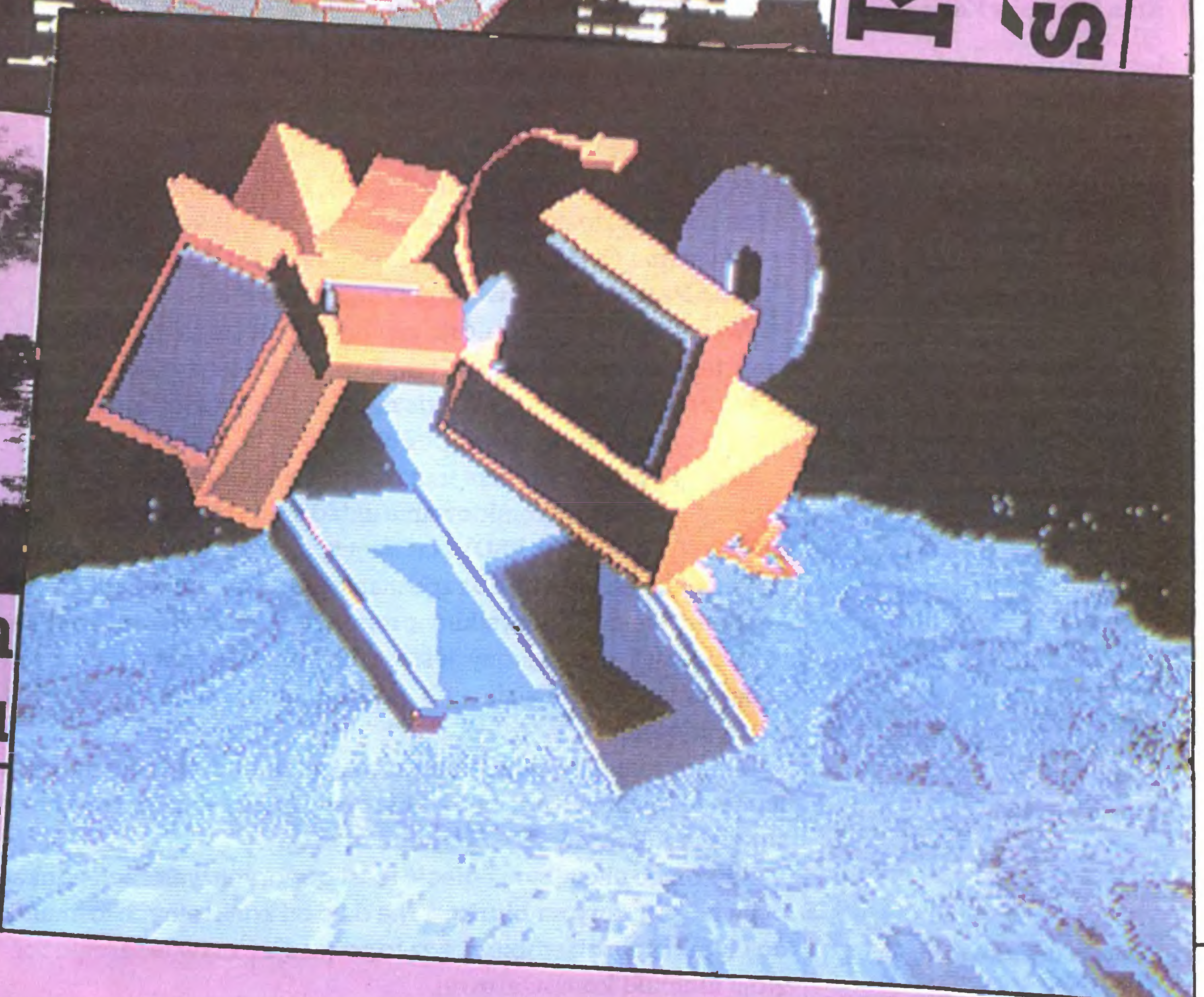
Komputer 10

\październik 1988\
popularny miesięcznik informatyczny:\#(31)'88\ cena 180 zł.

Kreowanie światów





Łaty na Turbo Pascalu

Zmiany na biurku

Kurier

3 Statystyczna firma
Grzegorz Eider

4 Na 10 dni przed drukiem

5 Kreowanie światów
Stanisław Marek Królak

7 Gary Kildall
(Postaci mikroświata)

7 Kronika
Ej-Bi-Em

8 Nowości

8 Terminator terminologiczny [16]
Stanisław Marek Królak

9 Komputeryzujemy się

9 Czytaj!

10 Nowe sieci
Piotr Fijałkowski

11 Listy

Komputer w domu

12 Pomiar wielkości fizycznych
Jacek Mielcarski

13 Funkcje BDOS

14 Łaty na Turbo Pascalu
Mariusz Pietruszka

15 Turbo Pascal - dziur w całym ciąg dalszy
Jacek Gwiazdka

16 Mikroprogramy dla Atari XE/XL
Tomasz Mazur

17 Kompletny katalog dysku
Artur P. Chmielewski

19 Poke n,∞
Grzegorz Czapkiewicz

21 Klub Mistrzów Komputera
Adam Nowicki
Leszek Rudak

22 Forum

Komputer w pracy

25 SuperKey
Dorota Stabrowska,
Kazimierz Lewartowski

28 Praktyka programowania (Projekt programu)
Adam Nowicki

29 Programowanie współbieżne [2]
Wiktor B. Daszczuk

31 Dziury w całym - raz jeszcze

32 Ujarzmianie myszki [3]
Roland Wacławek

34 Abel
Krzysztof Kontek

38 Olivetti PC1
Zenon Rudak

39 Drobiazgi
Zenon Rudak

41 Zmiany na biurku
Zenon Rudak

42 Prosto z dysku

Mikromarket

43 Sprzedam spółkę!
Władysław Majewski

44 Wnioski (nie)uwzględnione [3]
Władysław Majewski

45-55 Ogłoszenia

56 Giełda

Szef kuchni poleca:



Październik zgodnie ze starym zwyczajem rozpoczyna nowy rok akademicki. Do tej tradycji dostosowaliśmy się i my - przygotowaliśmy numer nieco poważniejszy, co nie znaczy nudniejszy.

Trudno wyrokować o własnych tekstach i nie zamierzam tego robić, a jeśli proponuję Państwu przeczytanie **"Kreowanie światów"**, to ze względu na tematykę i osobę mojego rozmówcy. Profesor *Marek Hołyński* z Massachusetts Institute of Technology jest uważany za jednego z najlepszych specjalistów na świecie w dziedzinie grafiki komputerowej.

Komputer C 128 to dzisiaj w większości krajów z pewnością zabawka dla przedszkolaków. Tymczasem w części sygnowanej ***"W domu"*** proponujemy zastosowania tego komputera do raczej poważnych zadań. Czy ta ryzykowna propozycja podoba się Czytelnikom? - nie wiem. Myślę jednak, że jest to ciekawa oferta dla tych, którzy mają mały komputer i nie bardzo wiedzą jak go wykorzystać.

Licznych Czytelników zainteresują zapewne ulepszenia i poprawki w kompilatorze Turbo Pascala. Propozycje zmian w wersjach przeznaczonych dla komputerów pracujących pod nadzorem systemu CP/M zawierają teksty **"Łaty na Turbo Pascalu"** i **"Turbo Pascal - dziur w całym ciąg dalszy"**.

Właścicieli ST zainteresują zapewne procedury przedstawione w tekście pod tytułem **"Kompletny katalog dysku"**.

Poza tym w tej części naszego miesięcznika znajdą Państwo jak zwykle stałe rubryki - a więc **"Poke n, ∞"**, **"Forum"**, **"KMK"**.

Blok ***"W pracy"*** otwiera materiał poświęcony językowi wysokiego poziomu do projektowania układów o programowalnej logice (skrót angielski PLD). Nazwa języka - **"ABEL"** - jest jednocześnie tytułem tekstu, do zapoznania się z nim zachęcam nie tylko specjalistów. Wszystkim proponuję przeczytanie artykułu **"SuperKey"**, poświęcony jest on bowiem jednemu z licznych programów umożliwiających podniesienie komfortu pracy z komputerem.

W tym numerze wracamy do rozpoczętego w kwietniowym numerze tekstu **"Programowanie współbieżne"** oraz kończymy serial zatytułowany **"Ujarzmianie myszki"**.

Na koniec chciałbym polecić tekst o sporych walorach edukacyjnych **"Zmiany na biurku"**. Nie daje on konkretnych formułek co i w jaki sposób zrobić, ale za to pokazuje trendy i kierunki rozwoju techniki komputerowej.

Życzę przyjemnej lektury i zapraszam za miesiąc.

Stanisław Marek Królak

10 (31)

"Komputer" Popularny Miesięcznik Informatyczny - pismo miłośników i użytkowników mikrokomputerów redagują:

Marek Młynarski (red. nacz.)
Grzegorz Eider (z-ca red. nacz.)
Władysław Majewski (z-ca red. nacz.)
Stanisław M. Królak (sekr.red.)
Marek Car (publicystyka)
Grzegorz Czapkiewicz (programy)
Mariusz Dec (sprzęt)
Zenon Rudak (sprzęt)
Tomasz Zieliński (listy)
oraz zespół:
Zbigniew Blewoński, Andrzej Kadlof, Tomasz Mazur, Wiesław Migut, Wojciech Olejniczak, Juliusz Rawicz, Leszek Rudak, Jakub Tatarkiewicz, Roland Wacławek (Katowice) i współpracownicy: Maciej Borkowski (Poznań), Tadeusz Jedynak (Tarnowskie Góry), Jarosław Kania, Zbigniew Kasprzycki, Marek Matuszczak, Adam Nowicki, Mariusz Pietruszka (Tarnowskie Góry), Tadeusz Wilczek, Andrzej Załuski (Kraków)

Redakcja graficzno-techniczna:
Stefan Szczypka (kier.)
Małgorzata Łuzińska
Piotr Kakiet
Magdalena Stachorzyńska
(operatorka komputera)

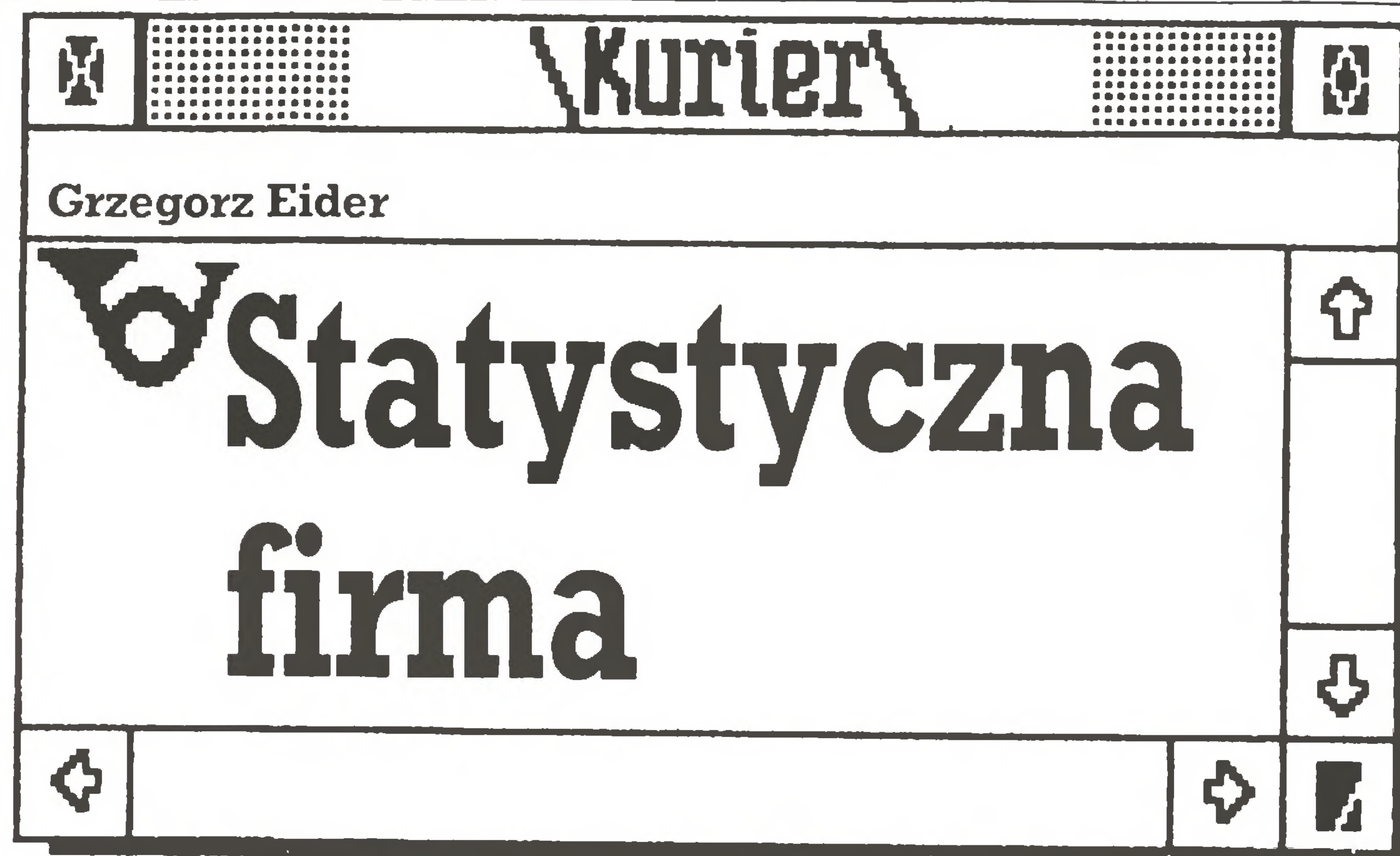
Redakcja programów komputerowych:
Jerzy Pusiak – kier.
Leszek Gołębiowski
Krzysztof Matey
ul Koszykowa 6A
00-564 Warszawa
282201 w. 312

Korekta: Maria Omiecińska,
Romualda Miarecka

Wydawca: Warszawskie Wydawnictwo Prasowe RSW „Prasa-Książka-Ruch", Al. Jerozolimskie 125/127, 02-017 Warszawa, tel. centrali: 28-52-31.
Redakcja: ul. Koszykowa 6A, 00-564 Warszawa, tel. 28-22-01 w. 243 lub 290 telex 813230 csdk pl
Skład i druk: Prasowe Zakłady Graficzne,
Łódź, ul. Armii Czerwonej 28.
Cena: 180 zł. Zam. 2752/88, U-32.

Prenumeratę od instytucji przyjmują oddziały RSW, a od osób prywatnych poczta (na wsi także doręczyciele). Cena prenumeraty rocznej 2160 zł, półrocznej 1080 zł, kwartalnej 540 zł. Prenumeratę ze zleceniem wysyłki za granicę (droższą o 50% dla osób prywatnych i o 100% dla instytucji) przyjmuje Centrala Kolportażu RSW, ul. Towarowa 28, 00-958 Warszawa, I PKO BP XV Oddz. W-wa 1658-201045-139-11.

Nakład 115 000 egz.
Nr indeksu 36-345 ISSN 0860-2514
Dyskietkę z tekstami do numeru przekazano do składu w dniu 13.07.1988.



Miesiące wakacyjne niezbyt sprzyjają zakrojonej na szerszą skalę działalności. Rynek komputerowy (jak wiele innych) nieco "przygasa", wiadomo - urlopy. Mimo to redakcji udało się w tym okresie uruchomić bazę danych o firmach komputerowych. KATALOG FIRM KOMPUTEROWYCH - tak bowiem nazywa się baza danych - zawiera informacje adresowe o 250 firmach. Część z tych firm zdecydowała się zamieścić w bazie swoje oferty - użytkownik znajdzie w niej także kilkadziesiąt nazwisk ludzi związanych z rynkiem komputerowym.

W wyniku niefortunnego błędu redakcyjnego (Czytelników raz jeszcze przepraszamy) w numerze 7/88 nie ukazał się "Mikromarket", w którym zapowiadaliśmy edycję bazy KFK, dlatego też pozwolę sobie przypomnieć jak dostać własną kopię bazy: można ją skopiować (dyski 5,25" IBM) raz w tygodniu (czwartek 10^{00}-12^{00}) w redakcji lub kupić u jednego z dealerów (jesteśmy w trakcie podpisywania umów z firmami na terenie całego kraju). Baza jest dostępna także w redakcyjnej sieci FIDO.

Dane zgromadzone w bazie pozwalają lepiej poznać rynek komputerowy w naszym kraju. Oczywiście w bazie nie są ujęte wszystkie firmy (może czwarta lub nawet jedynie piąta ich część), mimo to wydaje się, że dysponujemy wystarczająco wiarygodną próbką, by móc pokusić się o pewne uogólnienia i wnioski. Tezę tę uzasadnia fakt, iż w bazie znalazły się firmy uczestniczące w największych targach i wystawach mijającego roku oraz prowadzące intensywną kampanię reklamową. Z dużym zatem prawdopodobieństwem możemy powiedzieć, iż nie brakuje w bazie żadnej aktywnej, liczącej się firmy.

Geografia polskiego biznesu komputerowego nie jest skomplikowana. W dwunastu wielkich miastach koncentruje się 77% firm komputerowych. W rzeczywistości koncentracja jest jeszcze większa, bowiem wiele firm mieści się w suburbiach wielkich ośrodków (np. Marki czy Zielonka dla Warszawy). Największe skupiska firm to (w nawiasach procent w stosunku do wszystkich firm w bazie):

1. Warszawa wraz z suburbiami - (42,9%)
2. Trójmiasto - (13,2%)
3. Poznań wraz z suburbiami - (9,6%)
4. Region katowicki - (9,2%)

Jak widać, w tych czterech centrach skupionych jest 75% polskich firm komputerowych. Zauważmy, że również tam odbywają się wszystkie najważniejsze imprezy tego rynku (np. warszawski "Computer" czyli wystawa Agpolu, poznański "Infosystem", "Baltsoft" na Wybrzeżu i śląski "Softarg").

Właściwie struktura tego rynku nie jest zaskakująca. Ponieważ miażdżąca większość firm zajmuje się handlem (jeśli nie wyłącznie, to przynajmniej w części), i to handlem z użytkownikami, zrozumiałe jest, że firmy komputerowe powstają tam, gdzie są duże skupiska potencjalnych odbiorców. Stąd koncentracja.

Dziwić może stosunkowo nieduży udział rynku aglomeracji śląskiej. Sądzę, że w tym wypadku ma miejsce pewne zafałszowanie wynikające z faktu, iż jest to chłonny rynek, a w konsekwencji firmy na nim działające nie muszą agresywnie pokazywać się na terenie całego kraju.

Pozostawmy geografię na boku i przypatrzmy się kapitałowi i jego organizacji. Najchętniej wykorzystywaną przez menedżerów tego rynku formą organizacyjną jest spółka z ograniczoną odpowiedzialnością - spośród wszystkich firm 47,6% to sp. z o.o. Nie są to - wbrew potocznym wyobrażeniom - jedynie firmy prywatne. Kapitał zaangażowany w tych spółkach to:

- prywatny – 82% spółek,
- państwowo-prywatny – 8% spółek,
- państwowy – 7% spółek,
- spółdzielczy – 2% spółek,
- spółdzielczo-państwowy – 1% spółek.

Konieczne jest tutaj podkreślenie z naciskiem, iż mówimy o liczbie firm, a nie o wskaźnikach ekonomicznych. Gdybyśmy tabelę tę zestawiali przykładowo dla wielkości kapitału albo obrotów, sytuacja mogłaby wyglądać odmiennie (niestety, firmy nie chcą ujawniać swoich tajemnic finansowych nikomu poza fiskusem, nie dysponujemy więc takimi danymi).

Drugą kategorię (41,9% firm) stanowią przedsiębiorstwa. Nie jest to grupa jednorodna, bowiem tworzą ją zarówno duże "klasyczne" przedsiębiorstwa państwowe, jak i niewielkie prywatne firmy, które w dawnej nomenklaturze prawnej należałoby nazwać warsztatami rzemieślniczymi. Wydaje się, że między innymi dzięki rynkowi komputerowemu (choć głównie dzięki krachowi gospodarczemu i wymuszanym reformom) kategorie "rzemieślnik" czy "warsztat rzemieślniczy" zostaną wreszcie odłożone do lamusa średniowiecznej historii, a zastąpią je nieco współcześniejsze - "przedsiębiorca", "biznesmen", "przedsiębiorstwo", "firma". Dość jednak dywagacji. Przedsiębiorstwa państwowe stanowią w tej grupie 36,4% firm. Pozostałe to przedsiębiorstwa prywatne (krajowe i zagraniczne). Owe przedsiębiorstwa zagraniczne to firmy polonijne, polonijno-zagraniczne i zagraniczne. W subtelności tego podziału zagłębiać się nie będziemy - tak czy inaczej kapitał tych przedsiębiorstw to "twarda" waluta.

Trzecią kategorię firm stanowią spółki akcyjne (5,7% wszystkich firm). Jest ich niemal dziesięciokrotnie mniej niż spółek z ograniczoną odpowiedzialnością, mimo że S.A. wydają się być atrakcyjną formą organizacyjną. Prezes jednej ze spółek z o.o. powiedział mi, iż rozważał wariant spółki akcyjnej, strona prawna wydała mu się jednak zbyt niejednoznaczna (jego prawnikowi zresztą też). Nie wszyscy jednak są tego zdania i spółki akcyjne powstają.

Zamykają naszą listę spółdzielnie stanowiące 4,8% firm działających na rynku komputerowym.

Dotychczas mówiliśmy jedynie o sprawach niejako organizacyjnych, abstrahując od rodzaju działalności prowadzonej przez firmy. Z natury rzeczy informacje te trudniej poddają się obróbce statystycznej. W formularzu skierowanym do firm z bazy danych znalazły się jednak i rubryki dotyczące ich aktywności. Oto jak widzą swoją działalność firmy-respondenci:

- 82% firm produkuje oprogramowanie,
- 62% firm prowadzi doradztwo,
- 60% firm produkuje sprzęt komputerowy,
- 52% firm prowadzi szkolenia,
- 44% firm prowadzi naprawy,
- 40% firm prowadzi zlecony serwis,
- 38% firm zajmuje się pośrednictwem,
- 16% firm prowadzi działalność wydawniczą.

Dane te trzeba traktować z przymrużeniem oka. Są one - oględnie mówiąc - niezbyt prawdziwe, po części w wyniku niezrozumienia poszczególnych kategorii działalności, po części w wyniku świadomego fałszerstwa. I tak bez wątpienia pośrednictwo (skup od prywatnych dostawców i sprzedaż) prowadzi więcej firm, niż fakt ten zadeklarowało, mniej zaś prowadzi produkcję sprzętu, w której to właśnie kategorii ukryte jest pośrednictwo. Widać handel na zasadzie: kupiłem, dałem własną nalepkę i kabel sieciowy, by wreszcie sprzedać z zyskiem - wydaje się naszym przemysłowcom zajęciem niezbyt godnym i wolą raczej mówić o produkcji.

Jest raczej mało prawdopodobne, by niemal połowa firm prowadziła serwis zlecony przez jakiegoś wytwórcę czy firmę wysyłkową (nieporozumienie wynika zapewne z niedostrzegania słówka "zlecony"). I wreszcie 62% firm prowadzących konsulting to ewidentne nieporozumienie. Cóż to bowiem za konsultant, który sam dostarcza sprzęt i oprogramowanie - z góry wiadomo, że będzie usiłował wcisnąć klientowi swoje produkty.

Dane dotyczące rodzajów działalności traktować zatem należy raczej jako projekcję wyobrażeń o firmie niż obraz rzeczywistej aktywności.

Gdybyśmy chcieli na podstawie przedstawionych danych stworzyć obraz "statystycznej" firmy, byłaby to zapewne prywatna spółka z ograniczoną odpowiedzialnością (lub prywatne przedsiębiorstwo) mieszcząca się w Warszawie (ewentualnie w Trójmieście). Firma ta handlowałaby komputerami dołączając do sprzętu również oprogramowanie (zapewne jakiś f-k lub gospodarka materiałowa), w swoich ulotkach i reklamach wmawiając klientom, że najpierw przeanalizuje potrzeby użytkownika, a dopiero później zaproponuje najbardziej adekwatne (czyli własne) rozwiązanie sprzętowo-programowe.

Na 10 dni przed drukiem

• Wkrótce:

Listopad

Targi INFORMACJA'88 - 14-17.11., Katowice, hala "Spodek"

VI Seminarium Centr.Szk.Inf .ZETO Łódź "Oprogramowanie narzędziowe i użytkowe, wielodostęp" - 21-24.11., Karpacz

II Forum Firm Komputerowych PTI i KUMP - 24-26.11., Olsztyn

Grudzień

V Jesienna Szkoła PTI - 5-11.12, Mrągowo

Seminarium sekcji CAD/CAM KUMP "Systemy Rotringa" - 9.12., Warszawa, siedziba PZ PROMAX, Bartycka 26 paw. A.

Konferencja "Mikroprocesory w elektrotechnice" - 14-15.12., Dom Technika, Łódź, Pl. Komuny Paryskiej 5a (organizator: Instytut Informatyki PŁ, ul. Piotrkowska 220, 90-496 Łódź)

Styczeń

IV Międzynarodowe Targi Komputer'89 - Warszawa, 24-27.01.

W rubryce "Wkrótce" publikujemy informacje o ważniejszych pokazach, targach, konferencjach i seminariach powiązanych z techniką mikrokomputerową, na które zostaliśmy pisemnie zaproszeni z sześciotygodniowym wyprzedzeniem.

• Z kroniki:

IV Konferencja "Informatyka w szkole" (28-30.IX.) w Wałbrzychu stała się areną dyskusji między "Elwro" i pracownikami oświaty na temat przydatności "Elwro 800 jr" oraz słabego rozwoju oprogramowania edukacyjnego (w katalogu, rozprowadzanym przez Zakład Edukacji Komputerowej OBR Pomocy Naukowych i Sprzętu Szkolnego w Warszawie, ul. Śniadeckich 17, figurują tylko 42 pozycje, a wiele z nich jest przestarzałych).

"Atari w szkole" to hasło wystawy MikroExpo'88 w Muzeum Techniki. Pokazano na niej interesującą sieć małych Atari, ale i widoczny był brak oprogramowania edukacyjnego.

Zlot Użytkowników MSX odbył się w IX Liceum im. K.Hoffmanowej w Warszawie. Pokazano wiele ciekawych programÓw. Dzięki Składnicy Harcerskiej komputery MSX mogłyby być już dziś oświatowym, bogato oprogramowanym standardem.

Pierwsze 20 lat informatyki polskiej - w 40 lat od powstania Grupy Aparatów Matematycznych przy Państwowym Instytucie Matematycznym odbyła się sesja zorganizowana przez PTI i SEP. W trakcie sesji prof. Leon Łukaszewicz przyznał, że pierwszy polski funkcjonujący komputer XYZ wzorowany był na IBM 701, a dla odmiany prof. Antoni Kiliński przypomniał, że w latach 1937-39 prof. Janusz Groszkowski prowadził pionierskie prace, przerwane z uwagi na zbliżającą się wojnę, które mogły doprowadzić do powstania pierwszego w świecie komputera elektronicznego.

• Turbo-Pascal 5.0

W niecały rok po wersji 4.0 Borland zaanonsował TP 5.0, wzbogacony m.in. o zintegrowany debugger, emulację 8087, wykorzystanie pamięci EMS, Turbo Assembler oraz proceduralne typy, zmienne i parametry.

• Mikrolaur' 89

KUMP, PTI, Agpol oraz nasza redakcja po raz drugi organizują konkurs "Mikrolaur".

Prace krajowe prosimy zgłaszać do 30 listopada 1988 r. na adres KUMP (Warszawa, ul. Przemyska 11a), po uprzednim zapoznaniu się z regulaminem konkursu - można go otrzymać w tymczasowej siedzibie KUMP przy ul. Dantyszka 12 lub w naszej redakcji.

Sąd konkursowy przyzna medale i dyplomy "Mikrolaur'89". Ich rozdanie nastąpi podczas IV Targów "Komputer'89".

• Komputerowa resocjalizacja

Otwarciu (20.10.) Centralnego Klubu Komputerowego Młodych Mistrzów Techniki ZSMP towarzyszyło wręczenie nagród (komputerów) dla przodujących klubów MMT. Najlepsi - z wojewódzkiego klubu komputerowego MMT "Olimp" przy Zespole Szkół Ogólnokształcących w Sztumie (woj. elbląskie) - dotarli z pokazami nowej techniki nawet do miejscowego Zakładu Karnego.

CCK mieści się w Warszawie w klubie MSM "Starówka" przy ul. Jana Brożka 18, tel. 36-27-49.

• AutoCAD po polsku!

W trakcie targów oprogramowania "Softarg" w Katowicach odbył się przedpremierowy pokaz polskiej wersji językowej programu AutoCAD, która powstała w wyniku współpracy firmy Autodesk z jej autoryzowanym polskim dystrybutorem - spółką Aplikom z Łodzi. Zaprezentowano także najnowszą, 10 wersję programu, która trafi równocześnie na rynki zachodnie i polski (po polsku oczywiście) w styczniu 1989 r. - za ok. 8 mln zł. Pokazano także biblioteki polskich norm mechanicznych do wykorzystania w programie.

Pierwszym w Polsce legalnym użytkownikiem AutoCAD-a jest WSI w Radomiu, a następnymi będą zapewne Zelmer Rzeszów, Bipronaft Kraków i Polcolor Piaseczno.

• PL-Tekst 3.0

Przed miesiącem pytaliśmy o losy najnowszej wersji produktu firmy CSK. Pod koniec września podczas Ogólnopolskich Targów Oprogramowania "Softarg'88" PL-Tekst 3.0 został ostatecznie wystawiony, a przy okazji firma obniżyła cenę tego produktu.

• Acorn-klub

W Warszawie powstał Klub Użytkowników Mikrokomputerów Acorn-BBC i Acorn-Master korzystający z uprzejmości (udostępnienie sprzętu) kierownictwa warszawskich strażaków. Klub zaprasza użytkowników komputerów firmy Acorn na comiesięczne spotkania - zawsze w ostatnią środę miesiąca o godz. 18.00) w salach klubu Syrena, ul. Chłodna 4.

• Obok komputera - konkurs

Wojewódzki Klub Komputerowy MMiT i RW TMMT w Bydgoszczy wraz z PPH Polcomsoft Sp. z o.o. i Bydgoską Fabryką Kabli KABEL ogłosiły konkurs na przyrządy, urządzenia i przedmioty stanowiące "otoczenie" komputera. Warunkiem uczestnictwa jest wiek do 35 lat i nadesłanie oryginalnej pracy w formie rysunku (najlepiej technicznego) lub modelu wraz z opisem i informacją o autorze na adres Wojewódzkiego Klubu Komputerowego (Piotra Skargi 3, 85-018 Bydgoszcz) do 31 stycznia 1989 r. Dla zwycięzców przewidziano nagrody od 15 do 50 tys. z.ł

• Komputer dla szkoły

Ewa Nych i Lech Madeyski z Wrocławia zwrócili się do naszej redakcji z apelem: "(sprzęt 8-bitowy)....po kilku latach traci całkowicie dla przedsiębiorstwa wartość użytkową i stoi w magazynach. Proponujemy podjęcie akcji przekazywania tego sprzętu nieodpatnie szkołom, z równoczesnym zwróceniem się do Ministerstwa Edukacji Narodowej o poparcie, a do ministra finansów o nie stosowanie wobec ofiarodawców negatywnych konsekwencji finansowych."

Do sprawy wrócimy wkrótce.

• Prenumerata

Przypominamy o prenumeracie "Komputera" na 1989 r. - termin wpłat upływa 10 listopada 1988 r.

• Za miesiąc w "Komputerze":

Test:	Schneider EUROPC - w domu 16 bitów zamiast 8
	Modem akustyczny DATAPHONs 21d-2
Sprzęt:	Sony SMC-210 - brakujące ogniwo
	Commodore Amiga i Amiga 2500
	Amstrad 6128 na serio
Programy:	Wirusowa gorączka - raport
	Polski wirus
	Liczby całkowite w Turbo Pascalu 4.0
	Niesforny kursor
Zastosowania:	
	Komputer dla medyka
	Komputer w studio
	Komputer w pracowni fotograficznej
Wywiad:	Z Markiem Hołyńskim o grafice - część 2
Reportaż:	Softarg'88
Mikromarket:	
	Katalog Firm Komputerowych - raport 1
	Bank Informacji Programowych BIS - raport 2

FIDO 21-19-85 od 17.00 do 8.00

"Na 10 dni przed drukiem" przygotował 25.10.88 Władysław Majewski.

Skład: ®LOGOSCRIPT z pomocą programu Ventura Publ. i drukarki brother HL-8.

\Kurier\

Stanisław Marek Królak

Kreowanie nowych światów

Z Markiem Hołyńskim, profesorem Massachusetts Institute of Technology i Boston University, rozmawia Stanisław Marek Królak.

W jakim kierunku idą prace czołowych laboratoriów w dziedzinie grafiki komputerowej - czy ciągle chodzi o poprawienie jakości obrazu, o jeszcze lepsze parametry, czy też doszliście już do pewnego kresu? Jakie są ograniczenia?

Nie ma żadnych ograniczeń. Obecnie są trzy główne kierunki uderzenia w zaawansowanej grafice komputerowej. Jeden to realizm wizualny, drugi - interakcja i trzeci - systemy specjalizowane do różnych zastosowań.

Każdy z tych kierunków to temat na osobną rozmowę. Co Ty na to?

Za trzy tygodnie wyjeżdżam. Nie damy rady.

To inaczej. W tej rozmowie ograniczymy się tylko do zagadnień związanych z realizmem wizualnym. Pozostałe działy badań nad grafiką moglibyśmy omówić w czasie następnej rozmowy, tuż przed Twoim powrotem do USA.

Zgoda.

Zacznijmy może od pytania ogólnego: Na jakim etapie rozwoju znajduje się grafika komputerowa służąca do przedstawiania rzeczywistości?

Realizm wizualny przeszedł już przez wszystkie podstawowe szczeble: najpierw opanował perspektywę i transformacje perspektywy, czyli to, co jest określane jako dwa i jedna druga wymiaru (dwa wymiary plus perspektywa), a następnie obraz trójwymiarowy.

Dzisiaj już standardowo grafikę robi się w trzech wymiarach, co oczywiście oznacza wprowadzenie dodatkowych danych, gdyż doszła oś Z. Nie koniec na tym - wszystkie transformacje również muszą obejmować trzy wymiary. Mimo że transformacje są zwykle robione macierzowo, ilość przetwarzanej informacji jest tu olbrzymia, potrzebne są więc pamięci o ogromnych pojemnościach oraz sprawne i szybkie komputery do jej przetwarzania. Ale jeśli chodzi o tworzenie obrazu rzeczywistego, to praktycznie problemy zostały już rozwiązane i możemy bez trudu tworzyć dowolne obrazy istniejące w rzeczywistości.

Powstałe trójwymiarowe cyfrowe modele obrazów możemy następnie rzutować na ekran na wiele sposobów, uwzględniając dowolne (również zmienne) położenie widza. Można usunąć płaszczyzny i krawędzie, które są niewidoczne, pokryć wszystkie obiekty powierzchniami o dowolnych stukturach, kolorach itp. (można korzystać tu z całych bibliotek powierzchni).

Ile kolorów macie do dyspozycji?

Jest ich bardzo dużo: w dużych systemach korzystamy z kilku milionów kolorów, w małych jest ich zwykle 256. Z tym że tylko pozornie jest 256, ponieważ najczęściej spośród nich tylko 8 jest predeterminowanych. To znaczy tylko 8 podstawowych: czerwony, zielony, niebieski i trzy ich kombinacje oraz biały i czarny jest zdeterminowanych i zapisanych w tabeli. Pozostałe 248 jest determinowanych zwykle przez użytkownika za pomocą funkcji "kolor", która ma trzy argumenty. Każdy z tych argumentów jest intensywnością czerwonego, zielonego i niebieskiego. Ponieważ te trzy kolory wyczerpują całe spektrum i różne intensywności tych trzech barw (kombinacja trzech), pozwalają na uzyskanie każdego koloru, w zasadzie użytkownik może sobie zaprogramować każdy kolor. Tak że ograniczenie do 248 jest pozorne i nie ma znaczenia, bo w każdej chwili można wymazać jedne i wprowadzić inne barwy.

Tym niemniej we wszystkich katalogach handlowych liczbę kolorów podaje się jako istotny parametr komputera.

Jest to tylko chwyt reklamowy. Powtarzam, że problem obrazu barwnego nie istnieje i mówienie o liczbie kolorów jako parametrze świadczącym o jakości systemu, a tym bardziej osiągnięciach technicznych, ma niewielki sens praktyczny.

Następnie udało się, po wielu próbach, zaprogramować gładkie powierzchnie. Aparat matematyczny np. krzywe Besiera czy tzw. *splines* pozwalają nie tylko aproksymować krzywe czy powierzchnie za pomocą odcinków linii prostej - choćby najmniejszych, ale zawsze odcinków - albo małych powierzchni poligonalnych, gdzie powierzchnie nigdy nie są do końca zgrabne, ale pozwalają już tworzyć pełne powierzchnie bez żadnych zadziorów na nich.

Trzecim znakomitym pomysłem, który dzisiaj jest bardzo pomocny w modelowaniu zjawisk naturalnych, jest geometria fraktalna. Daje ona możliwość wiernego modelowania zjawisk naturalnych.

Na łamach "Komputera" podejmowaliśmy kilkakrotnie tę tematykę, nigdy jednak nie udało się nam wytłumaczyć korzyści, które niosą ze sobą fraktale. Jakie zatem nowe możliwości daje ten aparat matematyczny w porównaniu z innymi sposobami przedstawiania rzeczywistości?

Jest to fundamentalna sprawa. Posłużę się przykładami. Do tej pory nie mieliśmy żadnego opisu, który pozwalałby na reprezentowanie płatków śniegu, krzaków, drzew, tęczy, wzgórza czy łańcucha górskiego. Owszem mieliśmy aparat, za pomocą którego opisywaliśmy te obrazy, ale było to klasyczne kartezjańskie podejście, które mówiło, że wszystko można zrobić z kostek, kul, cylindrów itd. Mało jest jednak rzeczy, które z kostek i piramid da się złożyć. Oczywiście nawet bardzo skomplikowane figury były opisywane - za pomocą równie skomplikowanych funkcji, często niejawnych.

Okazuje się jednak, że zjawiska naturalne mają pewną charakterystyczną cechę: są regularne, ale zarazem zakłócone w pewien sposób przez przypadek. Co to oznacza? Np. jeśli stworzymy łańcuch górski według najlepszych wzorów i wyświetlimy go, będzie widać, że jest to dzieło sztuczne, bo łancuch górski nie jest regularny, mimo że ma cechy regularne, które go wyodrębniają z innych obiektów w świecie rzeczywistym.

Geometria fraktalna wprowadza nową klasę obiektów, które są obiektami międzywymiarowymi i pozwala na realizowanie subtelnych zjawisk naturalnych.

Geometria klasyczna nie dawała takich możliwości?

Klasyczna geometria poszerzona została o obiekty, które są między wymiarami, obiekty, które są liniami, ale zajmują płaszczyznę. Tak jak na przykład niektóre dywany Sierpińskiego albo krzywa Kocha czy krzywa Hilberta. To znaczy jest to niewątpliwie linia, wiemy o tym, ale z drugiej strony ta linia jest nieskończona, choć ograniczona.

Przyzwyczailiśmy się, że linia wychodzi z minus nieskończoności, przechodzi przed nosem i kończy się w plus nieskończoności. Nie, linia nieskończona może się zawierać na ograniczonym obszarze i być nieskończona mimo to. Co więcej, ta linia nieskończona może zajmować powierzchnię tego obszaru i może być mierzona za pomocą tej powierzchni. Z jednej strony jest mierzona za pomocą długości, z drugiej - powierzchni. Są wzory na obliczenie wymiarowości tej linii i ta wymiarowość jest liczbą między 1 i 2, różne linie mają różne wymiary. Podobnie z obiektami o wymiarach między 2 i 3. Stąd się bierze nazwa fraktal, bo są to obiekty, które mają część rzeczywistą i frakcyjną (ułamkową) wymiaru.



Co więcej, geometria fraktalna jest bardzo podatna na wprowadzanie przypadków losowych. Wprowadzasz linię, która generalnie powtarza się, ale kolejne powtórzenia różnią się od siebie w pewnych detalach. Modelujesz łańcuch górski: każda nowa góra jest górą, ma szczyt, ma stoki itp., ale jest trochę inna niż wszystkie poprzednie góry. Modelujesz wybrzeże morskie i to, co widzisz jest ewidentnie wybrzeżem, ale każdy jego kawałek jest inny. Te kawałki mogą różnić się tak drobnymi detalami, że trudno je zdefiniować, ale różnią się. Są to na pozór subtelności bez znaczenia, ale właśnie one decydują o tym, że obraz nie wydaje się być sztucznie zsyntezowany, lecz naturalny.

Krótko mówiąc, geometria fraktalna to jedno z ważniejszych narzędzi tworzenia grafiki na najwyższym poziomie. Narzędzie opanowane czy dziedzina stale się rozwijająca?

Nie, nie jest to narzędzie opanowane do końca, ciągle pojawiają sie nowe rzeczy i nowe pomysły. Zwłaszcza dużo skorzystali na tym matematycy. Stosują oni wzory fraktalne Manderbrota albo wzory Julii na płaszczyźnie zespolonej do prześledzenia, jak zachowują się iteracje pewnych funkcji matematycznych. Do tej pory nie było to możliwe. Te dziedziny miały swój początek stosunkowo niedawno i są niesłychanie obiecujące.

Czy nadzieje, jakie wiążecie z fraktalami, wynikają z faktu, iż jest to najpewniejsza droga dojścia do wiernego obrazu świata naturalnego?

Fraktale to jedna z rzeczy poprawiających rzeczywistość obrazu, ale są inne niezwykle interesujące i niezbędne w dziedzinie zwanej realizmem wizualnym. Jedną z nich jest modelowanie światła. Jest już wiele funkcji modelujących źródło światła. Modelują one również, w jaki sposób światło się załamuje, odbija i przechodzi przez obiekty. Możesz na przykład modelować kule różnie oświetlone, z których jedne odbijają, a inne przepuszczają światło. Albo na przykład patrzysz przez jedną kulę na drugą, widzisz deformacje spowodowane przejściem światła przez różne kule itp. Może to być oczywiście obraz animowany, a więc ruchomy, kule mogą się przesuwać, obracać itd. Możesz analizować światło przechodzące przez obiekty, odbijające się od nich i trafiające do widza. Jest tu wiele innych możliwości. Są to modele szalenie interesujące, ale i bardzo skomplikowane.

> 6

5 <

Po co to wszystko? Symulujesz rzeczywistość i co dalej? Jakie są przykładowe konkretne zastosowania?

Owszem, służy to do symulacji rzeczywistości, ale nie tylko. Na przykład niedawno Japończycy zrobili film, który pokazuje różne sposoby oświetlenia pokoju. Masz pokój, różne źródła światła, możesz je włączać w dowolny sposób, widzisz jak się zmienia oświetlenie tego pokoju w ciągu dnia i w nocy. Gdy Księżyc świeci lub jest w nowiu. Widzisz co się dzieje, gdy jest burza z piorunami, jak rozkładają się cienie, jak rozkładają się kolory przedmiotów w zależności od oświetlenia itp.

Tego typu opracowania mogą być wykorzystywane w pracach badawczych, ale też i chyba do bardziej praktycznych celów?

Oczywiście, że tak. Wszystkie badania, nawet podstawowe, prowadzi się z myślą o tym, że będą wykorzystywane na co dzień w najróżniejszych dziedzinach. I są wykorzystywane. Poza tym pamiętaj, że my musimy zarobić na swoje utrzymanie.

W każdym razie od jakichś trzech lat jesteśmy w stanie wyświetlić na ekranie wszystko. Nie ma sytuacji spotykanych w rzeczywistości, których nie da się pokazać na ekranie. Można wyświetlić każdy obraz z taką samą dokładnością i dokładnie takimi samymi kolorami, oświetleniem, powierzchniami i rozdzielczością jak w rzeczywistości. W przypadku złożonych obrazów wymaga to czasu, dużej maszyny, oprogramowania, ale nie ma tu problemu nie rozwiązanego. I nie jest to symboliczne przedstawianie rzeczywistości, do czego ludzie się przyzwyczaili i co kojarzy im się z grafiką komputerową - np. w grach komputerowych człowiek miał kwadratową głowę i to wystarczało. Tego już nie ma, dzisiaj taka grafika po prostu nie istnieje. Jeśli przedstawiamy świat realny, to mamy wierny jego obraz.

Nie rozumiem jednego: mówisz, że można tworzyć obrazy o takiej samej dokładności i rozdzielczości, z jaką spotykamy się w świecie rzeczywistym. Przecież natura jest nieporównanie doskonalsza od najlepszego obrazu, a Was demiurgów obrazu krępuje chociażby jakaś graniczna rozdzielczość...

Niezupełnie. Jeśli chodzi o rozdzielczość, ograniczeniem jest wspomniane wcześniej 4096 x 4096, co przewyższa rozdzielczość Twojego oka. Gdy widzisz taki obraz na ekranie, to masz wrażenie, że jest on ciągły, nie jesteś w stanie odróżnić sąsiednich punktów, czyli masz to samo wrażenie, co przy oglądaniu obrazu rzeczywistego. Jeśli zrobisz fotografię obrazu rzeczywistego i obrazu komputerowego, będą to identyczne fotografie.

W zeszłym roku Sąd Najwyższy USA wydał orzeczenie, iż dokumentacja fotograficzna nie może być dowodem w sądownictwie amerykańskim. Dlaczego? Dlatego, że jesteśmy w stanie stworzyć za pomocą komputera obrazy i sytuacje, które w rzeczywistości nie miały miejsca. Zrobić tak, że za pomocą żadnych technik nie można rozstrzygnąć czy jest to zdjęcie rzeczywistości, czy też obraz syntetyczny, sztuczny.

Jakie maszyny są potrzebne do stworzenia tak doskonałego obrazu?

W zasadzie można zrobić to na każdej dużej maszynie. Dobrze mieć jak największą ze względu na liczbę bitów na piksel oraz szybkość.

Dałem tu przykład fotografii, obrazu statycznego, ale to samo można robić z obrazem dynamicznym, także w czasie rzeczywistym. I wówczas szybkość maszyny i jej wielkość ma decydujące znaczenie. Jako przykład można podać wszystkie symulatory lotu samolotów czy rakiet, kiedy trzeba w ciągu 1/30 czy 1/60 sekundy przetworzyć obraz i wyświetlić nowy. A składa się on z milionów punktów, które muszą być przeliczone od nowa. Tu używa się dużych maszyn np. typu Cray. Natomiast jeśli nie zależy Ci na czasie i program może "chodzić" bez przerwy przez dwa czy trzy dni, w zasadzie każda maszyna to zrobi, np. PC. Jedynym problemem będzie ekran, który to później wyświetli i pojemność pamięci, żeby schować ten program i zakodować wszystkie kolory.

Czy symulatory to szczytowe osiągnięcia grafiki w obszarze realizmu wizualnego?

Nie. Na jego czubku są programy animacyjne, w których jest jeszcze sporo do zrobienia, mimo że osiągnięcia są tu olbrzymie. Ale nie wszystko możemy jeszcze uzyskać.

Na przykład czego nie jesteście w stanie zrobić?

Są liczne programy do animacji różnych obiektów, ale nie możemy dotąd w sposób niezauważalny dla widza animować ruchu człowieka. Są oczywiście programy animujące człowieka, są całe filmy animowane, w których ludzie się poruszają, ale ten ruch zawsze mniej czy bardziej jest wyczuwalnie sztuczny.

A ruch innych obiektów?

Robimy bez problemu - zależy on od równań obiektu, które trzeba zastosować i wykonać odpowiednie przeliczenia. W przypadku człowieka problem polega na tym, że nie udało się stworzyć dotychczas pełnego opisu ruchu człowieka. Trzeba to robić empirycznie, umieszczając czujniki na ciele człowieka, przenosząc dane do komputera itp. Tak więc animowanie poruszającego się człowieka to zadanie do zrobienia.

Znowu okazuje się, że człowiek jest istotą bardziej złożoną niż sądzili niektórzy filozofowie?

Darujmy sobie filozofów, natomiast to, co powiedziałem nie oznacza, że jest to problem niewykonalny już dziś. Pewne próby zakończyły się pomyślnie. Istnieje już program, który na szkielecie ludzkim ma dokładne odwzorowanie dynamiki chodu i skoku z dwóch nóg. Jest to bardzo krótki film zrobiony w Ohio State University przez Dawida Zeltzera. Szkielet idzie, odbija się z dwóch nóg, ląduje, podnosi się, idzie dalej w sposób zupełnie naturalny (o ile można mówić o naturalnym sposobie poruszania się szkieletu).

Człowiek piętą achillesową animacji...

Nie zapominaj, że nawet tu grafika komputerowa daje zupełnie nowe możliwości. Przypomnij sobie klasyczną animację. Są tam dwie grupy fachowców: do pierwszej należy artysta najwyższej klasy, który robi główne sceny: rysuje pierwszą klatkę, po czym tę, która będzie po 15 sekundach. Do drugiej zaliczają się wyrobnicy, którzy rysują pozostałe 300 klatek pomiędzy scenami narysowanymi przez artystę. I największą pracą w animacji jest rozpisywanie obrazu na klatki, bo trzeba namalować te 24 klatki na sekundę, które dają wrażenie ciągłego obrazu. Komputer jest tu niezwykle użyteczny, bo można zakodować obie klatki namalowane przez artystę, po czym wydać polecenie, że transformacja tych dwóch obrazów ma być transformacją liniową rozpisaną na 300 klatek. Może się okazać, że transformację opisuje jakaś kwadratowa zależność, może jakaś inna, może być to jakieś bardzo skomplikowane równanie ruchu, które spowoduje, że wszystkie punkty zostaną miękko przeniesione w następny obraz, ale jakościowo nie ma to znaczenia i komputer tę pracę wykona.

Rozmawiamy o ograniczeniach, które wynikają raczej z niedostatków aparatu matematycznego, a nie niedoskonałości techniki komputerowej.

I z tego, i z tego. Nie wszystko da się opisać matematycznie ściśle, ale jak wspomniałem, dane można uzyskiwać empirycznie. Ten sposób jest już wykorzystywany nawet w reklamach telewizyjnych. Na przykład zastosowano go w popularnej reklamie telewizyjnej pt. "Sexy robot". Była to grafika telewizyjna przedstawiająca robota-kobietę, który siedzi w fotelu, obraca się, wyciąga rękę i wskazuje na puszkę z konserwami. Było to świetnie zrobione. Ale twórcy tej grafiki nie mieli oczywiście równań ruchu. Dlatego zatrudnili modelkę, która wykonywała dokładnie ten sam ruch i zapisali trójwymiarowo położenia wszystkich punktów ciała modelki. Otrzymali model empiryczny i wykorzystali uzyskane w ten sposób transformacje punktów do opisu swojego robota.

Podałem ten przykład, aby pokazać, że nie wszystko trzeba koniecznie w pełni formalizować, można się oprzeć na naturalnych opisach. Z tym że z jednej strony brakuje opisu matematycznego i nie wiadomo czy on w ogóle będzie kiedykolwiek dostępny (może będą biblioteki ruchów wzięte z natury?), ale z drugiej strony trzeba wykonywać ogromne obliczenia i nie wszystkie komputery mogą temu podołać. Nie we wszystkich maszynach można odwzorować pewne zjawiska naturalne ze względu na skomplikowanie modelu.

Doszliśmy więc do konkluzji, że jedynym poważnym problemem realizmu wizualnego jest animacja, a głównie poruszający się człowiek. Moglibyśmy na tym zakończyć naszą rozmowę, ale proponuję na koniec kilka problemów ogólniejszych.

Po co to wszystko? Działania, o których mówisz, zmierzają w gruncie rzeczy do stworzenia sztucznego syntetycznego świata, który byłby nie odróżnialny od świata realnego. Jaki może być kres tego dążenia - stworzenie urojonej rzeczywistości? Czy nie obawiasz się niebezpieczeństw stąd płynących? Nie są to może pytania do profesora nauk komputerowych, ale skądinąd wiem, że jesteś komputerowcem filozofującym.

Odpowiedź na postawione przez Ciebie problemy wymagałaby co najmniej osobnej rozmowy. Hasłowo odpowiem tak: Trudno nie obawiać się niebezpieczeństw, jakie otwierają komputery, w tym i zaawansowana grafika. Możliwości nadużyć jest tu wiele - od szantażowania męża czy żony poczynając, poprzez pokazanie polityka wygłaszającego przemówienie o dowolnej treści i w dowolnych okolicznościach, przemówienie, które nigdy nie miało miejsca, do prowokacji militarnych o nieobliczalnych wręcz następstwach.

To wszystko prawda, tylko czy świadomość, że możliwe są nieetyczne zachowania, ma nas powstrzymywać od twórczego doskonalenia siebie i świata? Na to pytanie trzeba odpowiedzieć przecząco, w przeciwnym razie człowiek skazany byłby na zejście do poziomu zwierzęcia.

Myślę, że korzyści płynące z rozwoju grafiki komputerowej są niepodważalne i tak znaczne, że wkrótce czeka nas rewolucja w wielu dziedzinach życia. Te przemiany już się rozpoczęły i dotykają obszarów, które jeszcze niedawno nie uznawały techniki komputerowej. Przykładem niech będą chociażby korzyści płynące z możliwości kręcenia filmów bez studia i dekoracji. Cały drugi plan można wygenerować. Więcej nawet - postaci aktorów mogłyby być animowane. Nie mówiąc o ponętnej perspektywie - by trzymać się sztuki - tworzenia światów, które nie istnieją. Ponętnej nie tylko technicznie, lecz i artystycznie.

Dzisiejsze wysiłki grafiki komputerowej podobne są do wysiłków malarzy XIX wieku. Najpierw starali się jak najwierniej przedstawić świat, a dopiero gdy to osiągnęli, zaczęli deformować rzeczywistość traktując te odkształcenia jako zabawę intelektualną, z której z czasem powstały nowe wartości.

Dzisiaj dziękuję. Za dwa tygodnie porozmawiamy o interakcji i zastosowaniach specjalnych.

Marek Hołyński, profesor MIT i Boston University, 41 lat, kawaler, ma narzeczoną i poważne zamiary. Studia ukończył na Wydziale Elektroniki Politechniki Warszawskiej. Pracował w Instytucie Maszyn Matematycznych, doktorat zrobił w Instytucie Cybernetyki PAN. Pociągała go filozofia i nauki humanistyczne. W 1974 roku zainteresował się grafiką komputerową. W latach 1967 - 1976 był współpracownikiem Tygodnika Studenckiego "Politechnik", a od 1972 do 1979 współpracował z "Polityką". Wydał trzy książki popularne: o grafice komputerowej, o sztucznej inteligencji i przeglądową o różnych intelektualnych inteligentnych humanistycznych zastosowaniach komputerów. Zapraszany na różne uniwersytety. W 1980 roku został zaproszony na MIT i od tej pory tam pracuje. Hobby: literatura, narciarstwo, żeglarstwo i gra Go. Komputera konkretnego nie poleca, bo o tym jaki komputer jest potrzebny, decyduje zastosowanie. W domu ma PC.

Kurier

Postaci mikroświata

Gary Kildall

Bardzo obszernie prezentujemy ostatnio system operacyjny CP/M. Jako uzupełnienie chcemy także przedstawić - obszerniej niż zazwyczaj - jego twórcę i początki.

Gary Kildall uzyskał w 1970 roku w University of Washington doktorat i przeniósł się do Pacific Grove (Monterey) w Kalifornii, gdzie został profesorem informatyki w Podyplomowej Szkole Morskiej. Lubił uczyć i atmosfera szkoły bardzo mu odpowiadała. Któregoś dnia na tablicy ogłoszeń spostrzegł kartkę: "mikrokomputer za 25$". Okazało się, że chodziło o mikroprocesor Intel 4004, ale w 1972 roku i tak była to okazja. Kildall kupił i za namową ojca, który szukał dla swej niewielkiej szkoły nawigacyjnej w Seattle urządzenia do obliczania trójkątów nawigacyjnych, napisał i uruchomił pierwsze programy arytmetyczne. Okazało się, że procesor miał istotne ograniczenia: był zbyt powolny i mało dokładny.

Rok później Kildall miał okazję obejrzeć wydział mikroprocesorów firmy Intel. Niebawem został jej konsultantem. Poznał 4004, grzebiąc w nim kilka miesięcy tak dokładnie, że w firmie uznano go za wariata. Był pewien, że już nie wróci do pracy z dużymi komputerami, lecz badając pierwszy 8-bitowy mikroprocesor Intela (8008) musiał stworzyć jego emulator na minikomputerze, by móc sprawnie tworzyć i testować przeznaczone dlań programy.

W ciągu kilku miesięcy stworzył język PL/M - mikroprocesorową wersję popularyzowanego przez IBM języka PL/I, znacznie bogatszego od jedynego dostępnego wówczas dla mikroprocesorów języka - Basica.

Jako częściowe wynagrodzenie za pracę otrzymał od Intela mały komputer. Umieścił go w swojej klasie, tworząc pierwsze w Szkole Morskiej laboratorium mikrokomputerowe. Gdy Intel zmienił w "Intellecu-8" procesor 8008 na 8080 i dał Kildallowi monitor oraz szybki czytnik taśmy perforowanej, Gary mógł ze swymi uczniami pracować na sprzęcie porównywalnym z Altairem (tak nazywał się pierwszy mikrokomputer wprowadzony na rynek w styczniu 1975 roku).

Kildall odrzucił dwa podstawowe elementy dużego systemu: taśmę papierową i dyski magnetyczne, chcąc zastąpić je małym dyskiem od Shugarta - 5,25". Napisał specjalny program, brakowało mu jednak nadal odpowiedniego sterownika. Próbował go zrobić, chciał także opracować interfejs łączący komputer z magnetofonem, okazało sie jednak, że zdolności inżynierskie Kildalla pozostają w tyle za jego pomysłowością i umiejętnością programowania.

Był koniec 1973 roku, gdy zdecydował się poprosić o pomoc kolegę ze studiów, wówczas już profesora University of California w Berkeley - Johna Torode, przekonując go, że "zrobimy dużo naprawdę dobrych rzeczy, gdy stacja zacznie działać".

Torode wziął się do pracy, a Kildall - sfrustrowany niepowodzeniami ze sprzętem - napisał w ciągu kilku tygodni w języku PL/M prosty system operacyjny, który nazwał Control Program/Monitor, czyli CP/M.

W tym czasie jego kolega Ben Cooper próbował zarobić pieniądze budując urządzenie do stawiania horoskopów (choć astrologia w ogóle go nie interesowała) i poszukiwał programisty. Z Kildallem



szybko ukończyli dzieło, które karmione 25-centówkami miało drukować w sklepikach San Francisco horoskopy. I choć ostatecznie maszyna astrologiczna okazała się wielką klapą, dla Kildalla była ona pierwszym praktycznym testem systemu CP/M. Programując ją bowiem stworzył sobie podstawowe narzędzia: asembler i odpluskwiacz (ang. *debugger*) oraz rozpoczął pracę nad edytorem. Ponadto opracował interpreter języka Basic.

W połowie 1974 roku Torode i Kildall mieli gotowy dyskowy system operacyjny. Do swej pracy byli jednak nastawieni sceptycznie, nie spodziewając się po niej zysków i traktując ją jako swojego rodzaju wprawkę: Kildall pisał programy, Torode zajmował się sprzętem. Sprzedali w San Francisco dwie maszyny firmie Omron, zanim na rynku pojawił się Altair. Interes prowadzili niezależnie: Torode budował komputery jako Digital Systems, a później Digital Microsystems, Kildall pisał programy jako Intergalactic Digital Research (później już nieco skromniej jako Digital Research). Firmą początkowo kierowała żona Kildalla Dorothy, występując pod panieńskim nazwiskiem McEwen (nie chciała być bowiem dla klientów "tylko" żoną Gary'ego). Ona także namówiła Kildalla, aby sprzedawać CP/M jako gotowy pakiet. Gary nie zdawał sobie w tym czasie zupełnie sprawy z wartości tego programu, który wcześniej - lojalnie - zaproponował jako Control Program for Microprocessors (lub CP/M-80) Intelowi. Ten ostatni nie okazał jednak zainteresowania. Początkowo (za 90$) prawo do korzystania z CP/M jako systemu operacyjnego wszystkich swoich produktów kupił Thomas Lafleur (GNAT Computers), a w roku 1977 IMSAI Manufacturing z San Leandro - za 25000$.

Firma Kildalla - Digital Research Inc. - rozwijała się bardzo szybko, zajmując się m.in. sprzedażą programów aplikacyjnych i do roku 1985 pozostawała w ścisłej czołówce producentów oprogramowania. Potem, wraz z wyparciem CP/M przez MS-DOS z roli czołowego systemu operacyjnego dla mikrokomputerów, nadeszły trudniejsze czasy: nie udało się wylansować programu GEM jako standardowego środowiska graficznego dla mikrokomputerów, systemu DOS+ jako pomostu między CP/M i MS-DOS ani Concurrent DOS jako standardowego środowiska wielodostępnego. Obecnie, w epoce przechodzenia IBM z MS-DOS na OS/2, Digital Research odzyskuje część utraconego terenu dzięki udanej i szybko wprowadzonej na rynek wersji Concurrent DOS 386.

W ostatnich latach Gary Kildall znalazł kolejne dodatkowe zajęcie - bardzo sprawnie i ciekawie prowadzi niezwykle popularny program "The Computer Chronicles" w TV Boston.

Opracowała **Danuta Majewska**

Kurier

Kronika

Laureaci INFOSEM'88

Kolejne spotkanie **INFOSEM** - Ogólnopolskiego Seminarium Studenckich Kół Naukowych Informatyki - odbyło się w połowie maja br., tym razem w Lublinie. Spośród dwudziestu kilku polskich KNI w imprezie uczestniczyła zaledwie jedna trzecia. Przyszłoroczny INFOSEM'89 ma być bardziej eksponowany[1].

Pierwsze miejsce w **sekcji** M - o nieco przydługiej nazwie Metody i środki informatyki oraz teoretyczne podstawy informatyki - zdobył za *Moduł obsługi biblioteki w systemie Turbo Pascal* - student WAT kol. **A. Kurdyła.**

Pierwsze miejsce w **sekcji Z** - Zastosowania informatyki - zdobył przedstawiciel Uniwersytetu Gdańskiego **A. Dżuryk** za *Komputerowe wspomaganie wielkości sprzedaży jako integralny moduł skomputeryzowanego systemu informacji rynkowej*. Referowany system sam niedawno wdrożył w amerykańskiej firmie budowlanej MASTIC, korzystając z międzynarodowego stypendium praktykanckiego AIESEC[2]. **ZŁOTY DYSK ELASTYCZNY '87** za działalność prowadzoną w ubiegłym roku zdobyło Koło Zainteresowań Cybernetycznych... oczywiście przy WAT!

Do zobaczenia wiosną 1989 na nowym INFOSEM-inarium!

Ej-Bi-Em

[1] Adres dla korespondencji: RADA KOORDYNACYJNA Studenckich Kół Naukowych Informatyki, Uniwersytet Szczeciński, Instytut Cybernetyki Ekonomicznej i Informatyki, Mickiewicza 66; 71-101 Szczecin.

[2] Association Internationale des Etudiants des Sciences Economiques et du Commerce - AIESEC - międzynarodowe stowarzyszenie studentów ekonomii i handlu.



Kurier

Nowości

System operacyjny dla Am29000

Dotąd mikroprocesory o architekturze RISC były stosowane w minikomputerach pracujących pod kontrolą systemu Unix w zastosowaniach typu CAD (projektowanie wspomagane komputerowo). Dlatego opracowany przez Advanced Micro Devices i Ready Systems system operacyjny VRTX32 stał się wśród informatyków sensacją. System ten przeznaczony dla Am29000 (dokładnie budowę tego mikroprocesora opisaliśmy w numerze marcowym) pracuje w czasie rzeczywistym - można go zastosować w sterownikach przemysłowych.

Projektanci zrezygnowali ze stosowania pamięci wirtualnej. Uznali, że układy pamięci są tak tanie a przestrzeń adresowa Am29000 tak duża, iż pamięć wirtualna nie jest potrzebna. Upraszcza to sposób dostępu do pamięci. Zrezygnowano również z wydłużającej czas dostępu do pamięci ochrony zasobów, pozostawiając programistom decyzję o sposobach ewentualnego zabezpieczenia pamięci przed nielegalnym dostępem.

Problemy sprawiła liczba rejestrów Am29000 - jest ich 192. Przyporządkowanie programowi użytkownika wszystkich wydłużyłoby czas ich odkładania na stos podczas zmiany wykonywanego zadania. Przypisanie zadaniu ograniczonej liczby rejestrów utrudnia pracę kompilatora. Wobec tego VRTX32 tworzy z części rejestrów szybki stos, na który, w razie potrzeby, jest odkładana zawartość rejestrów pozostałych - z tych użytkownikowi są udostępniane wszystkie.

Koprocesor arytmetyczny Am9027 może pracować niezależnie od Am29000. Wywołuje to opóźnienie - czekanie na zakończenie wykonywania instrukcji przez koprocesor - gdy zewnętrzne przerwanie powoduje zmianę zadania. VRTX32 sprawdza, czy nowe zadanie wymaga użycia koprocesora - jeśli nie, Am29000 zajmuje się już obsługą przerwania.

Obsługa zadań związanych z systemem operacyjnym pochłania około 40% mocy obliczeniowej procesora, co dla wielozadaniowego systemu pracującego w czasie rzeczywistym jest wynikiem dobrym.

(ADAN)

* * *

Forsa płynie...

Staramy się w naszych krótkich notkach pokazywać możliwie na bieżąco kondycję przemysłu komputerowego na świecie. Mamy cichą nadzieję, że być może w ten sposób zachęcimy także krajowych potentatów do publikowania swych wyników finansowych.

Początek roku stał w USA pod znakiem oszałamiających sukcesów firmy Apple, która pod hasłem "zamiast czekać na OS/2 kup oryginalnego Macintosha - my wszystko to mamy od dawna" po latach prób zdobywa wreszcie modelem Macintosh II świat biznesu.

W pierwszym kwartale br. obroty firmy osiągnęły 867,2 miliona dolarów - o 50% więcej niż przed rokiem, a zyski - 79,7 miliona - aż o 135% więcej niż rok temu.

Nieźle radzi sobie również Motorola, której sprzedaż wzrosła w pierwszym kwartale o 26%, do 1,95 mld dolarów, przy czym w dziale produkcji układów scalonych sprzedaż wzrosła o 25%.

Na tym tle za rozczarowujące uznano wyniki firmy DEC (Digital Equipment Company), której sprzedaż wzrosła tylko o 17%, do 2,82 miliarda dolarów, a zyski nawet spadły - z 307 do 305 mln dolarów.

O ile spadające zyski są wynikiem bardzo wysokich w DEC nakładów na rozwój techniczny, o tyle słaba dynamika sprzedaży dziwi, gdyż sojusz z DEC - jak się powszechnie uważa - stał się głównym czynnikiem napędowym sukcesu Apple i Macintosha II, a ponadto DEC jest największym na świecie producentem komputerów pracujących pod nadzorem systemu Unix, odnoszącego ostatnio nowe oszałamiające sukcesy.

Jak dalece pomocny może być Unix w walce o rynek, świadczą wyniki firmy Sun Microsystems, która zwiększyła sprzedaż swych graficznych stanowisk pracy (workstations) o 81% (do 259,7 mln dolarów w ciągu I kwartału) mimo narastającej konkurencji, jaką dla stacji klasy Sun stają się najwydajniejsze mikrokomputery epoki 80386 i transputerów.

Miarą stanu branży są wyniki IBM, który zwiększył sprzedaż o 10% - do 11,75 mld dol. kwartalnie, a zyski o 16% - do 913 mln kwartalnie. Sprzedaż komputerów (2/3 obrotów giganta) wzrosła o 14%, natomiast spadek obrotów odnotowały działy serwisu, wypożyczeń i obsługi.

WM

Nowości programowe

Summer'87 to oznaczenie niecierpliwie oczekiwanej najnowszej wersji kompilatora języka dBase III+ firmy Nantuncket Corporation. Oferuje on jeszcze większe możliwości oraz mniejszą (oby!) liczbę błędów.

* * *

Space Birds firmy Astronomical Data Service to program pozwalający określać położenie sztucznych satelitów Ziemi. Po podaniu długości i szerokości geograficznej punktu obserwacyjnego, jego wysokości nad poziomem morza oraz czasu, w jakim chcemy prowadzić obserwacje, program - korzystając z bazy danych NASA - informuje,.jakie satelity mamy szansę zobaczyć. (45$)

* * *

SEGS z Edmond Software Inc. to inżynierski program graficzny, pozwalający rysować ponad 5000 punktów dla każdej z 10 krzywych i max. 4 niezależnych osi y. Ma wbudowany arkusz kalkulacyjny pozwalający przygotować dane przed rysowaniem. Dane można wprowadzać z klawiatury lub czytać z plików ASCII przygotowanych innymi arkuszami. Rysunki mogą być drukowane na drukarkach mozaikowych i laserowych lub rysowane na ploterze. (195$)

* * *

Specific Fonts 4 (SF4) ze Specific Solutions dostarcza za 75$ użytkownikom Ventury i GEM Desktop Publishera narzędzia do pisania wyrażeń matematycznych oraz rysowania diagramów dołączanych później do pisanych dokumentów. SF4 zawiera niezbędne symbole i znaki matematyczne oraz litery greckie a także bibliotekę dekoracyjnych liter i cyfr dla uatrakcyjnienia sporządzonych wykresów.

opr. Z.B

Kurier

Stanisław Marek Królak

Terminator terminologiczny (16)

W tym felietonie zajmiemy się przypominaniem pewnych prostych zasad znanych jeszcze ze szkoły, ale - jak świadczą o tym teksty napływające do redakcji - dawno zapomnianych. Bardzo wiele problemów przysparzają skróty wyrazów lub związków wyrazowych. Do najczęstszych należy niepoprawna pisownia skrótowców używanych w innych niż mianownik przypadkach oraz brak kropek.

Sięgnijmy zatem do słowników języka polskiego. Znajdziemy w nich zasadę, że znane od dawna skrótowce odnoszące się do nazw pospolitych czytamy zwykle jako pełne wyrazy, piszemy je małą literą i pamiętać musimy, że **zawsze** stawiamy po nich kropkę - nawet w środku zdania. Przykładami tej grupy skrótowców są: np., itd., itp., jw., cdn., tzw. oraz pisany wielką literą skrót od Post scriptum - Ps. (przy okazji przypominam, że **nie stawiamy** kropki po skrótach typu dr, mgr).

Poczynając od XIX wieku pojawiają się nowe skrótowce i jest ich coraz więcej. Z nimi mamy też najwięcej kłopotów. Do tej grupy zaliczymy częste na naszych łamach skróty typu: DOS, RAM, ROM, PC. Główny problem, przed którym tu stajemy, polega na tym, że nie wiadomo jak **to** odmieniać.

Zasada ogólna obowiązująca w języku polskim głosi, że skrótowce literowe i głoskowe kończące się w wymowie samogłoską -a, -o, -u, -e pozostawiamy **zawsze w formie nieodmiennej.**

W odniesieniu do skrótowców literowych, głoskowych i mieszanych, które w całości są pisane wielkimi literami i które w wymowie kończą się na spółgłoskę, kierujemy się dwoma zasadami:

Pierwsza - nadrzędna - to zasada nieodmienności. Zgodnie z nią piszemy np. członkowie PAN, wydawnictwa MON, pliki DOS, w systemie GEM.

Druga zasada - stosowana w języku mówionym - polega na odmianie skrótowców według określonego wzoru deklinacyjnego. Jeśli decydujemy się stosować tę zasadę, pamiętać musimy, że **końcówkę fleksyjną piszemy małą literą i oddzielamy od tematu dywizem** - i wyłącznie dywizem, nie apostrofem, nie przecinkiem ani spacją czy brakiem znaku oddzielającego temat od końcówki. Poprawny jest więc wyłącznie zapis: członkowie **PAN-u**, wydawnictwa **MON-u**, pliki **DOS-u**, w **GEM-ie**. Niedopuszczalna pisownia na przykład taka: DOSu, DOS'u, DOSU.

To wszystko na dzisiaj. Opóźniona o tydzień ze względu na teksty do tego numeru podróż poślubna czeka. Państwu życzę równie przyjemnych zdarzeń w życiu.

\Kurier\

Komputeryzujemy się

Komputeryzujemy się, ale nie wszyscy. Dzieci pracowników "Siarkopolu" - nie. Jak dowiedzieliśmy się z artykułu Elżbiety Wierzbickiej w "Życiu Warszawy", dzieciom pracowników "Siarkopolu" w zbliżeniu z komputerami przeszkodziło warszawskie kuratorium.

"Siarkopol" mianowicie chciał urządzić w stolicy kolonie komputerowe. *"Głównym wabikiem i dla rodziców, i dla dzieci były komputery. Wyjście poza prościutkie gry, zajęcia z instruktorami, naukę programowania i dostęp do urządzeń, które nie tylko w Tarnobrzegu i okolicach są dziś dla ludzi synonimem postępu i nowoczesności - były gwoździem kolonijnego programu."* Organizacji kolonii podjęły się biura turystyczne "Flisak" i "Tebos", to ostatnie wyremontowało - na podstawie umowy z kuratorium - własnymi siłami jedną ze szkół warszawskich, żeby w niej umieścić kolonie. Ale na kilkanaście dni przed wyjazdem do Warszawy "Siarkopol" otrzymał teleks, że kolonie nie mogą się odbyć, bo "Tebosowi" cofnięto uprawnienia. W "Siarkopolu" nie bardzo się jeszcze tym przejęto - ostatecznie kolonie mogli zorganizować sami. Kiedy jednak przedstawiciele przedsiębiorstwa przybyli do stolicy, aby to wyjaśnić - okazało się, że wyremontowany budynek szkolny przekazano na całe wakacje komendzie OHP.

"Muszę pani powiedzieć, że po prostu oniemiałem, kiedy kurator Fedorowicz oświadczył, że (...) podpis dyrektora Moskały jest nieważny, bo on sam wyjechał za granicę. Przecież to kpiny! Mnie nie interesuje nazwisko pana M., tylko to, że był on urzędnikiem reprezentującym oświatę warszawską i jego zgoda dawała nam określone uprawnienia" - relacjonuje "Życie Warszawy" reakcję niedoszłych kontrahentów kuratorium. Zaproponowano im, żeby wyremontowali sobie teraz inną szkołę. *("Nie chcę być złośliwy, ale to wszystko wygląda tak, jakby warszawskie kuratorium znalazło sobie sposób na remonty szkół (...) A potem mogliby nas wymanewrować, tak jak zrobili z "Tebosem").*

W innym - podanym "Siarkopolowi" uzasadnieniu zerwania umowy akcentowano fakt, że na organizacji kolonii wzbogacą się pośrednicy, stosujący wygórowane ceny. *"Rozbawił mnie zarzut kuratora Polańskiego o zawyżaniu kosztów, bo "Tebos" chciał po 550 zł od łóżka, natomiast pani Kowalska z Dzielnicowego Zespołu Ekonomiczno-Administracyjnego Szkół przy ul. Paryskiej mówiła o 1000 zł i to w szkole, którą my mielibyśmy im wyremontować!"* - mówi przedstawicielka "Siarkopolu".

W końcu "Siarkopol" podsunął władzom oświatowym myśl, żeby do tej innej nieremontowanej szkoły przeprowadzić młodzież z OHP. *"Będzie dla nich robota na miejscu. Wtedy nasze dzieci miałyby kolonię w szkole, która dla nich została wyremontowana. Niestety, nie przeszło".*

*

Powyższa opowieść ma z tematem, którym nasza rubryka się zajmuje, tylko tyle wspólnego, że chodziło o "kolonię komputerową"; tak przynajmniej redaktorowi rubryki początkowo się wydawało. Postanowił jednak całą tę historię przytoczyć, kiedy w miesięczniku "Kontrasty" przeczytał następującą opinię prof. Witolda Marciszewskiego, kierownika Zakładu Logiki, Metodologii i Filozofii białostockiej filii Uniwersytetu Warszawskiego:

"Moim zdaniem te aparaty - komputery - trochę leczą z całego zła umysłowego, które latami narastało się w tym kraju. W systemie, który doprowadził do kryzysu, brakuje nam głównie dwóch rzeczy - odpowiedzialności i sprawdzalności. Jedno jest z drugim związane, bo jeżeli moje działania mają sprawdzalne skutki, to ja się wtedy czuję za nie odpowiedzialny. Jeżeli nie wiadomo, jaki zachodzi związek między decyzją ministra a pogorszeniem się losu emerytów, to ten minister odpowiedzialny nie będzie. A właśnie obsługa komputera, używanie komputera, wręcz obcowanie z komputerem radykalnie przestraja umysł ludzki."

*

Dzieci pracowników "Siarkopolu" wychowywane na deptaniu zobowiązań i niedotrzymywaniu umów oraz nie dopuszczane do oświaty komputerowej przez kuratorów oświaty i wychowania nie zyskały okazji, aby przestroić umysły w kierunku, którego życzy sobie prof. Marciszewski. Czy jednak profesor - entuzjasta komputerów, nie przecenia wpływu tych urządzeń na ludzkie postępowanie? Chyba nie. Oczywiście, nieodpowiedzialność niekoniecznie musi wynikać z braku logiki, braku rozumienia związków przyczyn ze skutkami. Doskonale można sobie wyobrazić nieodpowiedzialność opartą na logicznym rachunku. Kiedy instytucja działa w warunkach, w których MOŻE bezkarnie oświadczyć, że podpis jej przedstawiciela już jej nie zobowiązuje, bo dziś kto inny został dyrektorem - to oświadczy, jeśli tak jest jej wygodniej. Komputer wymusza logikę, ale nie wymusza etyki.

Natomiast "komputerowe myślenie" ułatwia obronę przed nieodpowiedzialnością. Pozwala na wyraźne ustalanie faktów nieodpowiedzialności i niesprawdzalności, o których mówi prof. Marciszewski. Komputer bowiem uczy stawiania pytań we właściwej kolejności i udzielania na nie jednoznacznych odpowiedzi. Albo kuratorium podpisało umowę, albo jej nie podpisało. Albo szkołę wyremontowano, albo nie wyremontowano. Albo dzieci pojechały na kolonie, albo nie pojechały.

Ludziom, którzy się do takiej dyscypliny rozumowania przyzwyczają, nie będzie można zapewne mącić w głowie, wciskać - zamiast owych jasnych odpowiedzi "tak" lub "nie" - kawałków o "głębokiej trosce", "ofiarnym wysiłku" i "dalszym doskonaleniu wypoczynku dzieci spędzających wakacje w Warszawie." Podobnie jak komputer będą na to całkowicie niewrażliwi.

JR.





\Kurier\

Czytaj!

Zbigniew Gutowski, Marian Molski "Komputer w pytaniach i odpowiedziach", WNT 1988, wyd. I, 49.700 + 300 egz., 277 str., 680 zł.

Rozwój informatyki i techniki komputerowej powoduje, że coraz szersze kręgi społeczeństwa chcąc nie chcąc będą musiały poznać podstawy tej dziedziny wiedzy - przynajmniej w stopniu pozwalającym na korzystanie z podstawowych zastosowań komputerów, takich jak redagowanie tekstu czy przekazywanie informacji za pomocą sieci komputerowych. Elementem ułatwiającym edukację jest ciągle żywe jeszcze zainteresowanie techniką komputerową. I chociaż w Stanach Zjednoczonych fascynacja tą dziedziną wyraźnie maleje, o czym świadczy coraz mniejsza liczba kandydatów na kierunki informatyczne czołowych uniwersytetów, to u nas rosnąca liczba komputerów zwiększa krąg potencjalnych czytelników książek z tej dziedziny. Starają się wykorzystać to wydawcy wypuszczając coraz to nowe poradniki, przewodniki i opracowania obliczone na masowego odbiorcę. W tej rubryce odnotowywaliśmy już takie pozycje.

Do tego typu książek należy też praca panów Gutowskiego i Molskiego. Przyjęta formuła pytań i odpowiedzi dobrze nadaje się do prowadzenia wykładu dla laików. Ułatwia swobodne poruszanie się w szerokim obszarze zagadnień, zwalniając od obowiązku ścisłego wyjaśniania pojęć i poruszanych zagadnień, a równocześnie dodaje wykładowi dynamiki zachęcając do lektury. W książce omówiono podstawowe pojęcia techniki mikrokomputerowej, strukturę komputera z dość szczegółowym przedstawieniem procesora i pamięci oraz systemy wejścia/wyjścia. Wiele uwagi poświęcono architekturze systemów komputerowych i oprogramowaniu. Można tu znaleźć podstawowe wiadomości np. o systemach wsadowych czy wielodostępnych, sieciach lokalnych, językach programowania, systemach operacyjnych czy programowaniu strukturalnym Dijkstry. Osobne miejsce poświęcono historii komputerów, ich ewolucji i najnowszym kierunkom rozwoju.

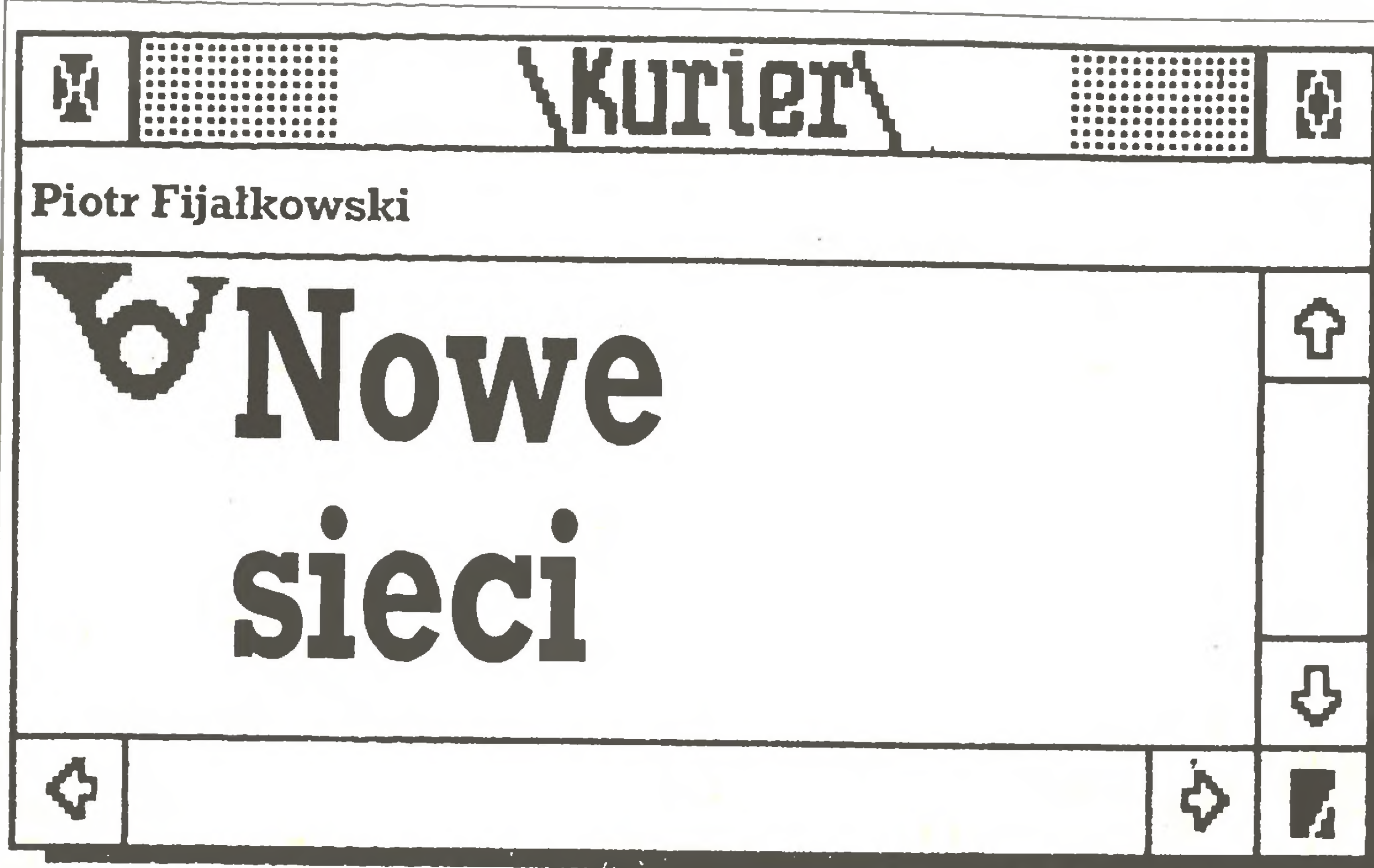
O ile o niektóre drobiazgi można się z Autorami spierać - np. o to czy uzasadnione jest poświęcanie większej uwagi jednym językom, a pominięcie lub ledwo wymienienie innych (C) - to błędem wydaje się pominięcie grafiki komputerowej oraz bardzo lakoniczne przedstawienie możliwości zastosowań komputerów osobistych.

Książka wydana jest dość starannie, ale na podłym papierze. Nieco do życzenia pozostawia też skorowidz, przydałby się w tego typu publikacji również słowniczek podstawowych terminów angielskich.

Autorzy nie zamieścili - niestety - wstępu, nie sposób więc ustalić, kiedy napisali książkę. Wiadomo tylko, że pracę oddano do składania 14.01.1986, a więc niemal trzy lata temu. W technice komputerowej to niemal wieczność.

S.M.K.





Piotr Fijałkowski

Nowe sieci

Między 30.05. a 2.06 br. w Düsseldorfie (RFN) odbyła się międzynarodowa konferencja dotycząca kierunków rozwoju i przyszłych zastosowań komputerowych, opartych o standard OSI i LAN MANAGER-a. Organizatorami konferencji były firmy Microsoft i 3Com.

Na konferencję zaproszono około 500 osób spośród licencjonowanych użytkowników systemu OS/2, w tym z Polski przedstawicieli firmy PZ Computex. Uczestnikami konferencji byli w większości pracownicy firm produkujących oprogramowanie użytkowe, a także małych i średnich firm produkujących sprzęt - między innymi z: 3COM, IBM, OMNINET, UNGERMAN-BASS, NOKIA etc. Prezentowali oni działanie LAN MANAGER-a w praktyce dla różnych realizacji sieci lokalnych.

Organizatorzy konferencji to dwie współpracujące ze sobą firmy, najbardziej zaawansowane w rozwoju oprogramowania sieciowego dla komputerów osobistych klasy IBM PC i PS/2. Są one jednocześnie współautorami LAN MANAGER-a, czyli implementacji na mikrokomputery osobiste coraz szerzej uznawanego standardu komunikacji pomiędzy komputerami OSI (OPEN SYSTEM INTERCONECTION).

Biorąc pod uwagę olbrzymi potencjał 15-18 mln już zainstalowanych mikrokomputerów osobistych, wymienione firmy postanowiły stworzyć możliwość lepszego jego wykorzystania poprzez wprowadzenie zunifikowanego standardu sieci.

Pojemności dyskowe i potencjał obliczeniowy drzemiące w mikrokomputerach są olbrzymie, lecz w chwili obecnej wykorzystane jedynie w niewielkim stopniu. Szacuje się, że co najwyżej 10% mikrokomputerów jest połączonych w sieć. Sieci te są użytkowane jedynie dla czerpania zasobów dzielonych, takich jak pliki czy urządzenia peryferyjne.

Ze względu na swoją uniwersalność LAN MANAGER pozwoli połączyć w nadchodzących 5 latach 30-50% mikrokomputerów. Korzystanie z sieci będzie niosło także nowe możliwości jakościowe, takie jak przetwarzanie rozproszone, czyli możliwość realizowania programu użytkowego na różnych komputerach sieci. LAN MANAGER dostarcza nowego typu zasobu dzielonego, jakim jest czas obliczeniowy wybranych komputerów w sieci. Nie jest wykluczone, że znaczenie rozwiązań programowo-sprzętowych, jakie prezentują firmy Microsoft i 3COM, jak również inne firmy pracujące w oparciu o standard OSI, może być tak rewolucyjne, jak było w swoim czasie np. wynalezienie telefonu. Gordon Letwin, autor książki "Inside OS/2", Microsoft Press, 1988, powiada, że każda rewolucja technologiczna składa się z 3 etapów:
- etapu "egzotycznej zabawki",
- etapu "ograniczonej użyteczności",
- etapu "absolutnej niezbędności".

Zdaniem wspomnianego autora, profesjonalne komputery osobiste i komunikacja pomiędzy nimi mają już za sobą 2 pierwsze etapy i wchodzą w etap trzeci.

Nowy standard sieci rozszerza możliwości wszystkich użytkowników sieci. Umożliwia on bowiem realizację trzech typów połączeń:
- mikrokomputer z innym mikrokomputerem w sieci LAN,
- LAN z inną siecią LAN,
- LAN z dużą maszyną cyfrową (MAINFRAME).

LAN MANAGER-a, korzystając z opublikowanych standardów, będzie można implementować na dowolny sprzęt. Pozwala on także na obsługę więcej niż jednego sterownika sieci, co umożliwi łączenie sieci o różnym charakterze (różne prędkości transmisji danych). LAN MANAGER jest użytkowo zgodny z istniejącymi systemami sieciowymi opartymi o standard NETBIOS i zapewnia łączenie mikrokomputerów pracujących z różnymi systemami operacyjnymi, tj. DOS, OS/2, Xenix.

Standard OSI, na którym bazuje LAN MANAGER, nie przekreśla dotychczas stworzonych protokołów komunikacji między komputerami, a umiejętnie wplata najbardziej udane z nich w swoją strukturę. Ma to kapitalne znaczenie dla rozwoju sieci komputerowych, gdyż zapewnia wprowadzenie nowych standardów drogą ewolucji, a nie metodą burzenia starych struktur i zastępowania ich nowymi.

Oprócz niezaprzeczalnych korzyści płynących z używania LAN MANAGER-a w sferze transportu informacji między komputerami, firma Microsoft dla zwiększenia atrakcyjności swojej oferty wyposażyła system OS/2 w wiele funkcji standardowych ułatwiających pisanie programów użytkowych w środowisku sieciowym. Są to funkcje kontroli dostępu do zasobów, funkcje sygnalizacji i kontroli dynamicznie zachodzących zdarzeń, a także funkcje pozwalające na przetwarzanie rozproszone. Na bazie tych systemowych możliwości zostały już stworzone narzędzia do produkcji programów aplikacyjnych, efektywnie wykorzystujących powstałe środowisko.

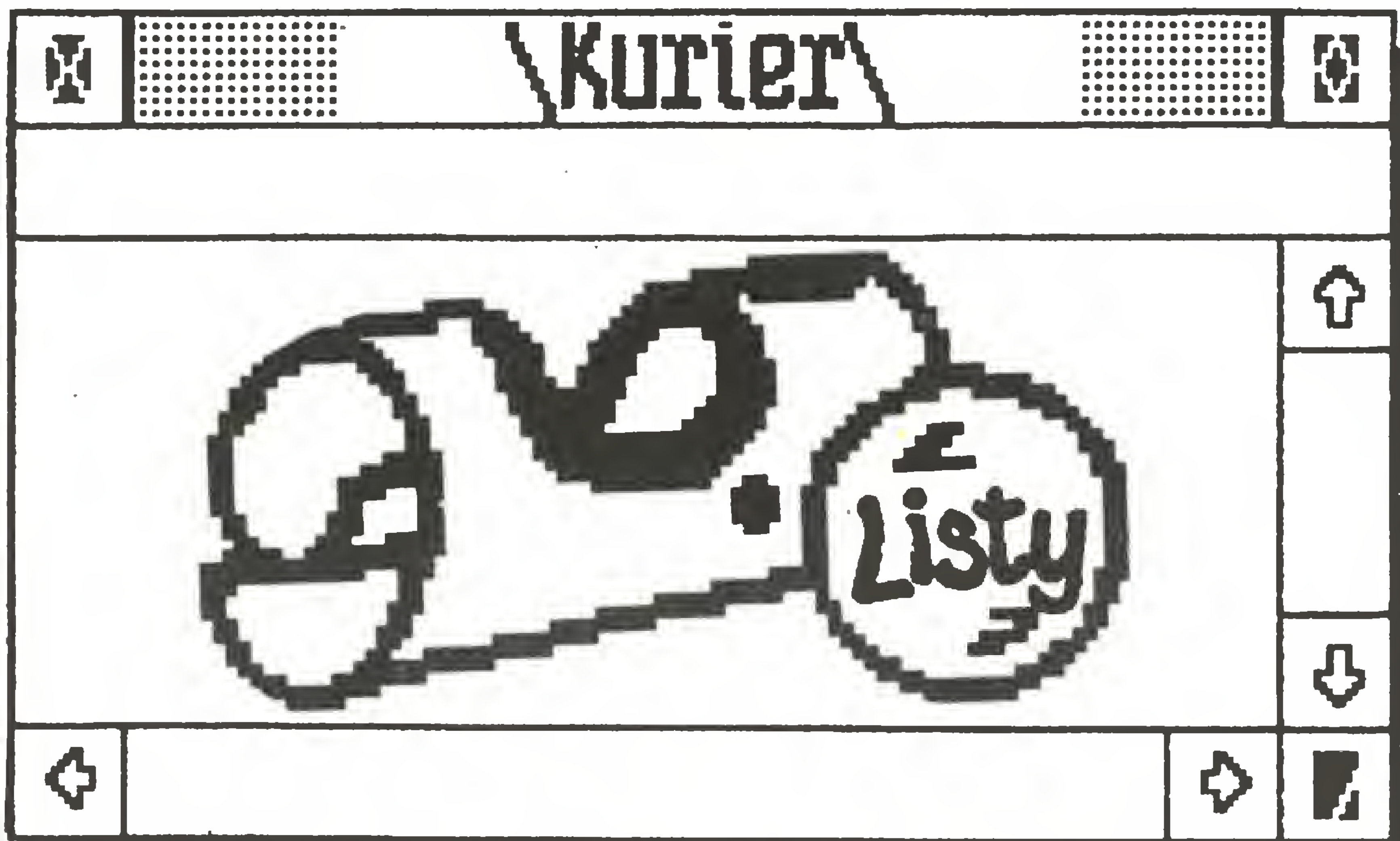
Jednym z najważniejszych produktów jest SQL SERVER, produkt firm Microsoft/Sybase/Ashton Tate. Zaproponowane przez producenta rozwiązanie umożliwia wirtualną prezentację dowolnej bazy danych obsługiwanej przez programy LAN MANAGER i SQL SERVER. Takie podejście pozwala na tworzenie uniwersalnych programów obsługujących dane (ang. *server*) dostępnych dla wszystkich użytkowników sieci. Nie zmusza posiadaczy baz danych do ich przeorganizowania, lecz umożliwia udostępnienie ich zawartości innym użytkownikom sieci. SQL SERVER/LAN MANAGER powinien rozwiać także wszelkie problemy wynikające z pracy w środowisku sieciowym. Zapewnia utrzymanie integralności baz danych poprzez standardowe mechanizmy blokowania dostępu do danych, *przetwarzanie transakcyjne* oraz archiwowanie zapisów w *transakcji*. SQL SERVER umożliwia rozbudowaną ochronę dostępu do danych.

Innym przebojowym narzędziem zaprezentowanym na konferencji jest kompilator RPC opracowany przez firmę NETWISE. Bazuje on na możliwości LAN MANAGER-a, polegającej na wykorzystaniu czasu obliczeniowego komputerów w sieci jako nowego rodzaju zasobu dzielonego. Umożliwia pisanie programów użytkowych w języku zbliżonym do C, dających możliwość aktywnego wykorzystania tego zasobu i zorganizowania "przetwarzania rozproszonego". Oznacza to, że niektóre procedury wywoływane w programie będą realizować się równolegle na innych komputerach w sieci. Efektywność takiego przetwarzania jest oczywista. Daje ono możliwość dokonywania szybkich obliczeń na wielu komputerach równocześnie. W przypadku przetwarzania tylko na jednym komputerze, z czerpaniem danych z innego, wielokrotnie zdarza się, że wybranie jednego rekordu z bazy danych wymusza transmisję wielu megabajtów danych poprzez sieć. Umieszczenie części programu użytkowego (zdalnej inteligencji) w komputerze właścicielu danych pozwala wyeliminować transmisję zbędnych informacji, a tym samym zwiększyć wydatnie jej efektywność i wydajność.

W warunkach polskich proponowane rozwiązania są mimo stosunkowo wysokich cen bardzo atrakcyjne. Należy jednak wziąć pod uwagę, iż system OS/2 jest systemem nowej generacji, co oznacza również bardzo wysoki stopień komplikacji, zbliżony do obserwowanego na tzw. dużych komputerach.

Jednym z wielu wniosków, jakie dla nas wynikają z konferencji, jest przykra prawda, że dotrzymanie kroku postępowi komputeryzacji wszystkich dziedzin życia wymagać będzie zwielokrotnionych nakładów finansowych na sprzęt, oprogramowanie i szeroko pojętą edukację. Pracownicy przyuczeni na kursach nie zastąpią wykształconych informatyków, a nic nie wiedząc o komputerach nie będziemy mogli używać ich w zakresie przygotowanym przez specjalistów.

I jeszcze jedna obserwacja. Skopiowane (czytaj ukradzione) oprogramowanie, szczególnie bez pełnej dokumentacji, jest bezwartościowe ze względu na swój stopień komplikacji. Produktem handlowym jest komplet: oprogramowanie, dokumentacja, szkolenie, konsultacje producenta a być może przede wszystkim szybki dostęp i możliwość zakupu po obniżonych cenach kolejnych, poprawionych i uzupełnionych wersji tego oprogramowania. I oto pojawia się zupełnie nowa jakość! Najświeższe poprawki produktów programowych są dostępne poprzez sieci telekomunikacyjne dla licencjonowanych odbiorców tego oprogramowania, czyli dla tych, którzy zakupili to oprogramowanie w sposób legalny. W ten sposób Microsoft udostępnia najświeższe informacje i poprawki dla licencjonowanych odbiorców OS/2 Software Development Kit i innych swoich produktów. A zasięg tej sieci informacyjnej jest ogólnoświatowy!



Firmy wysyłkowe na cenzurowanym

W związku z notatką ("Ceny sprzętu, firmy wysyłkowe", 4/88, str. 11) na temat ewentualnych nieprawidłowości w transakcjach z firmami wysyłkowymi reklamującymi się na naszych łamach informujemy, że otrzymaliśmy ostatnio parę skarg dotyczących działalności firmy Kolgar. Kopie listów naszych czytelników przekazaliśmy do właściciela firmy z prośbą o wyjaśnienie poruszanych w nich spraw. Po otrzymaniu odpowiedzi zrelacjonujemy je na naszych łamach.

Redakcja

* * *

"Uwagi" ciąg dalszy!

(Listy, 5/88, str. 13)

Chciałbym nawiązać do listu pana Krzysztofa Wesołowskiego z Poznania zamieszczonego w "Komputerze", który jest reakcją na mój poprzedni list. Poruszyłem w nim sprawę dwóch szeregów: sumującego i mnożącego.

Po pierwsze: bardzo się cieszę, że mój tekst wzbudził reakcję. Być może nasza wspólna korespondencja z panem K. Wesołowskim na łamach "Komputera" spowoduje wzrost zainteresowania matematyczną stroną informatyki i dyskusję na tematy algorytmizacji różnych problemów matematycznych. Braki w tej dziedzinie na łamach polskich czasopism komputerowych uważam za duży błąd i niedopatrzenie. Uważam, że niniejsza wymiana zdań może być bodźcem do zwiększenia ilości publikacji na ten temat.

Po drugie: chciałbym odeprzeć pewne zarzuty pod moim adresem, które wynikają z faktu niezrozumienia przez pana K. Wesołowskiego myśli przewodniej mojego poprzedniego listu. W żadnym miejscu nie napisałem, że oba programy są optymalne i że są wzorcami, które ktoś mógłby włączyć do swoich bibliotek podprogramów. Oba programy były praktycznie surowymi odwzorowaniami wzorów na te szeregi, nie próbowałem stosować jakichkolwiek algorytmów optymalizujących, programy liczyły wprost oba wzory (co być może wyglądało dość topornie). W programach tych chciałem zwrócić uwagę na problem natury czysto matematycznej, mianowicie na kwestię zbieżności przytoczonych szeregów, oraz na fakt, że program krótszy wcale nie musi być lepszy od innego, dłuższego (robiącego to samo). A zarzut "że nie wystarczy znać język programowania... " uważam za cokolwiek nieuzasadniony (zawód wyuczony - fizyk teoretyk, zawód wykonywany - programista).

Po trzecie: w pełni zgadzam się z panem K. Wesołowskim, że przed przystąpieniem do pisania programu warto jest stracić godzinę lub więcej na przemyślenie jak go napisać niż pisać od ręki, jak leci. Nie mam najmniejszego zamiaru polemizować z moim respondentem na temat wyboru odpowiedniego algorytmu do odpowiedniego zadania, gdyż całkowicie się z nim zgadzam (ale nie o to chodziło w moim poprzednim liście).

Gdyby na łamach "Komputera" powstała rubryka (co niniejszym postuluję) dotycząca rozważań matematycznych i optymalnych algorytmów dla problemów natury arytmetycznej, bardzo chętnie wezmę udział w dyskusji na jej łamach.

Z poważaniem

Włodzimierz Pszczółkowski
Poznań

* * *

Kyan Pascal - uzupełnienie

("Kyan Pascal - blaski i cienie", 6/88, str.15)

Szanowna Redakcjo!

Zostałem mile zaskoczony znajdując w "Komputerze" mój artykuł o Kyan Pascalu. W okresie od przesłania Wam tego tekstu pojawiły się jeszcze pewne szczegóły - już nie tak drażniące, ale swoiste dla Kyan Pascala. Pozwolę sobie podać malutkie uzupełnienie:

1. W pełni poprawnie działające instrukcje **PEEK** i **POKE** otrzymamy po deklaracjach:

```
FUNCTION PEEK (k:integer):integer;
VAR komorka:^char;
BEGIN
  ASSIGN (komorka,k);
  PEEK:=ord(komorka^)
END;
PROCEDURE POKE (k,l:integer);
VAR komorka:^char;
BEGIN
  ASSIGN(komorka,k);
  komorka^:=chr(l)
END;
```

2. Procedura **PAUZA** (str.16, wiersz 20 od góry) powinna zawierać znak # w linii przed słowem **END:**

```
BNE TUTAJ
#
END;
```

3. Instrukcja wiążąca **WITH** występuje, lecz jej użycie jest ograniczone do zmiennych typu:

```
VAR k:record i:integer;
             r:real;
       end;
```

Dla tablicy złożonej np. z takich elementów nie jest możliwe bezpośrednie użycie **WITH**. Inaczej mówiąc, dopuszczalne jest:

```
WITH K DO
BEGIN
  i:=1; r:=2.3
END
```

lecz dla tablicy A[...] zbudowanej z takich elementów już nie:

WITH A[1] DO - sygnalizuje błąd

BEGIN...

Serdecznie pozdrawiam

Witold R. Rudolf
Otmuchów

"Nie tylko do zabawy" - w praktyce (5/88, str. 18)

Szanowna Redakcjo!

Postanowiłem napisać ten list w związku z artykułem "Nie tylko do zabawy" pp. A. Izworskiego i R. Tadeusiewicza. Gorąco popieram propozycję autorów tegoż artykułu odnośnie publikacji serii miniaturowych programów w języku Basic z typowymi profesjonalnymi zagadnieniami.

Computer znaczy po francusku "liczyć, obliczać"; to samo znaczenie ma angielski wyraz compute.(...)

Pracuję w przemyśle, jestem mistrzem zmianowym i w związku z wykonywaną pracą mam dużo liczenia, zwłaszcza pod koniec miesiąca. Liczenie przy pomocy kalkulatorka zajmowało mi bardzo wiele godzin. Od ubiegłego roku korzystam z powodzeniem z Commodore 16, który ma przecież zaledwie 12 KB pamięci dostępnej dla Basica.

Początkowo napisałem kilka małych programów: obliczanie zarobku akordowego dla pracowników, obliczenia niezbędne dla prowadzenia dokumentacji produkcyjnej, sporządzanie grafiku pracy mistrzów zmianowych w systemie 4-brygadowym itp. Po każdej serii obliczeń musiałem wgrywać z kasety następny program, napisałem więc jeden obszerny, który zawiera menu i poszczególne podprogramy (które dawniej były oddzielnymi programami); zajmuje ok. 6 KB pamięci.(...)

Uważam, że inicjatywa pp. Izworskiego i Tadeusiewicza jest godna poparcia i bardzo pożądana.

Łączę pozdrowienia dla Redakcji

Wit-Antoni Szuberla
Wleń

* * *

Zaległe roczniki

Drogi Komputerze,

jestem Waszym czytelnikiem. Przez to, że słucham polskiego radia i trochę uczę się Waszego języka, mogę korzystać z Waszego pisma. Jesteście bardzo mi przydatni, bo tutaj u nas nie ma żadnego podobnego pisma.

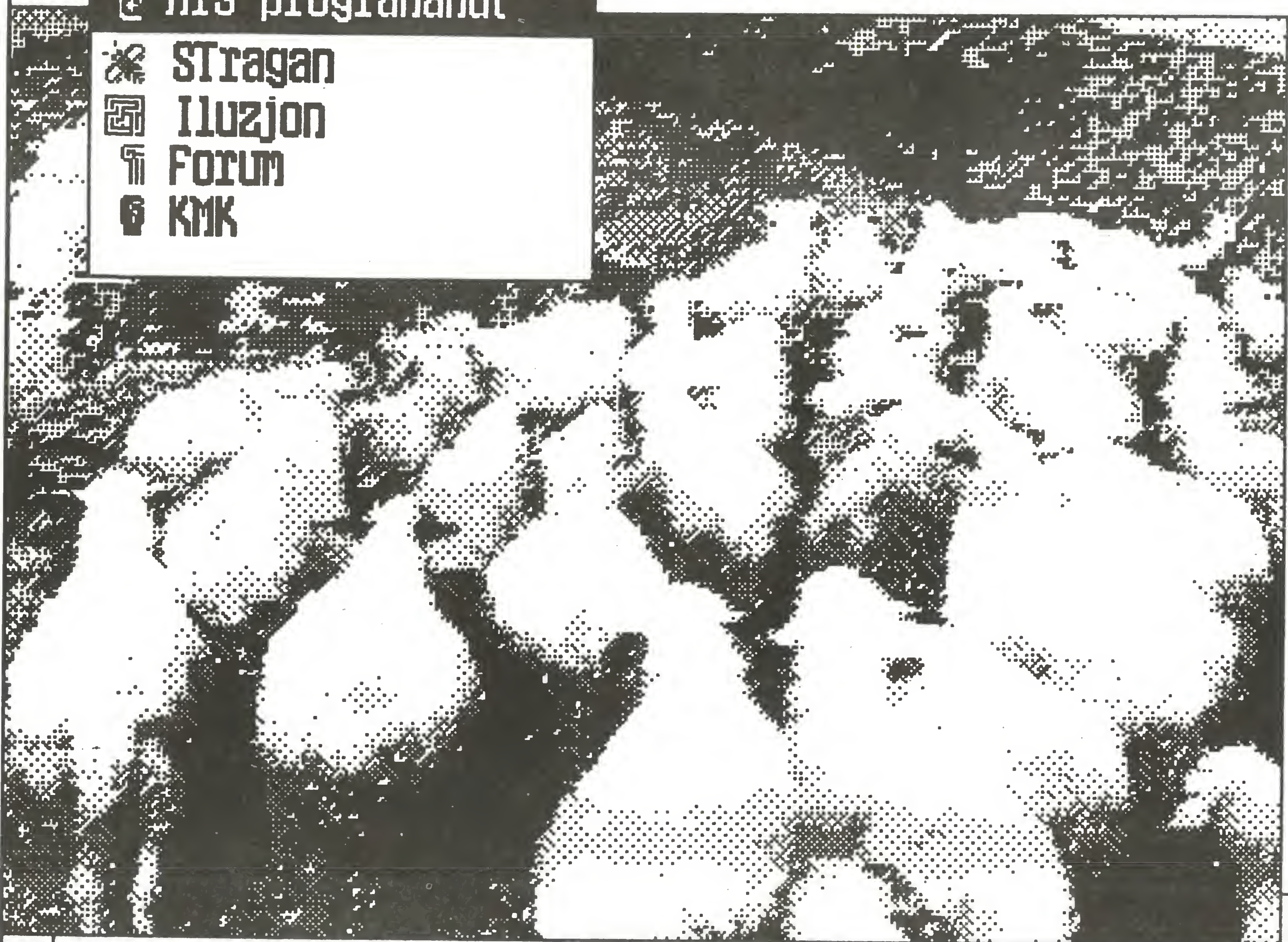
Niestety bardzo trudno jest dostać u nas "Komputer".

Proszę o wydrukowanie mojego listu w czasopiśmie, dlatego że chciałbym tą drogą zdobyć poprzednie roczniki "Komputera" (do 1988 roku) i "Bajtka" (do 1988 roku).

Bardzo dziękuję i pozdrawiam Was

Matias L. Kader
ul.Madonas 25-10
226084 Ryga 84
ZSRR





\W domu\

Jacek Mielcarski

Pomiar wielkości fizycznych na C128

Oprogramowanie dla komputera C 128 pracującego pod kontrolą systemu CP/M nie przewiduje użycia takich urządzeń jak joystick czy też manipulator kulkowy. W przypadkach wykorzystania tego komputera do przeprowadzania obliczeń lub przetwarzania danych (chodzi cały czas o pracę pod kontrolą CP/M) rzeczywiście nie jest to chyba potrzebne. Można jednak znaleźć inne zastosowanie dla komputera domowego, w którym jego komunikacja ze światem zewnętrznym staje się niezbędna.

Chciałbym zaproponować wykonanie prostego czujnika natężenia światła, temperatury lub innych wielkości fizycznych i oczywiście odpowiedniego oprogramowania. Komputer C 128 znakomicie nadaje się do tego celu, został bowiem od początku swego istnienia obdarzony przez naturę (a ściślej mówiąc przez konstruktorów) bogatym zestawem interfejsów. Dodajmy, że konstrukcja C 64 nie ustępowała C 128 pod tym względem ani trochę.

W sterowniki obrazu C 128 i C 64 wbudowane są cztery przetworniki mogące przetwarzać wartość dołączonego do nich opornika (potencjometru) na liczbę w zakresie od 0 do 255. Pracując w języku Basic - w podstawowym trybie C 128 - można wykorzystać do ich odczytu funkcję POT(n) (opis w instrukcji użytkownika). Dla systemu CP/M nie ma żadnych bibliotek, w których można znaleźć jakiś jej odpowiednik.

W instrukcji użytkownika przedstawione są opisy portów joysticka. Opisane tam wyprowadzenia:

POT AY - (5)
POT AX - (9)
+5V - (7)

i odpowiednio dla drugiego portu POT BX i POT BY są właśnie wejściami interesujących nas przetworników. Prosty schemat przedstawiony na rys.1. pokazuje metodę dołączenia fotoopornika do tego przetwornika.

Opornik R1 dołączony został jedynie w celu zabezpieczenia portu przed zwarciem. W rzeczywistości w komputerze przetwarzana jest nie tylko rezystancja R2, lecz stała czasowa RC opornika R2 i kondensatora C. Przy układzie zmontowanym jak na rys.1. - z opornikiem zabezpieczającym - możliwość uszkodzenia komputera praktycznie nie zachodzi. Układ można przetestować w Basicu 7.0 następującym programem:

```
10 PRINT POT(1), :GOTO 10
```

W momencie przysłaniania ręką fotoopornika wypisywane liczby powinny się zmieniać - przy prawidłowo dobranych elementach w zakresie od 0 do 255 (różnica pomiędzy całkowitą ciemnością i najjaśniejszym oświetleniem). Z braku fotoopornika można go zastąpić potencjometrem obrotowym lub suwakowym. Jeśli układ zachowuje się poprawnie, można przystąpić do pracy pod kontrolą systemu CP/M.

Oto przykłady odczytu rezystancji fotoopornika w asemblerze i Pascalu - ten drugi wynikać będzie bezpośrednio z pierwszego. Program został napisany w asemblerze Z80.

```
;          ******************************
;
;  .Z80
;
Port1    EQU 54297  ; Adres portu
                    ; pierwszego
```

```
                ; przetwornika.
Port:           ; Początek
                ; procedury
    Ld  BC,
        Port1   ; Umieszczamy
                ; adres portu
                ; w rejestrze BC
    In  A,(C)   ; Wczytujemy do
                ; akumulatora
                ; stan
                ; przetwornika
    Ret         ; Powrót z
                ; procedury - w
                ; akumulatorze
                ; znajduje się
                ; liczba opisująca
                ; stan
                ; przetwornika.
    END
; ******************************
```

Procedura ta pokazuje jedynie metodę dostępu do portu o określonym adresie - taki program nie spowoduje po uruchomieniu żadnego widocznego efektu. Nabiera ona praktycznego znaczenia dopiero po użyciu w konkretnym programie. Biorąc jednak pod uwagę, że obróbka danych eksperymentalnych jest łatwiejsza w języku wyższego rzędu niż w asemblerze, podaję odpowiednik tej procedury w języku Pascal (Turbo).

```
Function Paddle1 :Byte;
Var
  Value : Byte;

Begin
  InLine
  (
  $01/$19/$D4/ (* Ld bc,54297
               ($D419) *)
  $ED/$78/     (* In a,(c) *)
  $32/VALUE/   (* Ld (VALUE),a *)
  $00
  );
Paddle1 := Value;
End;
```

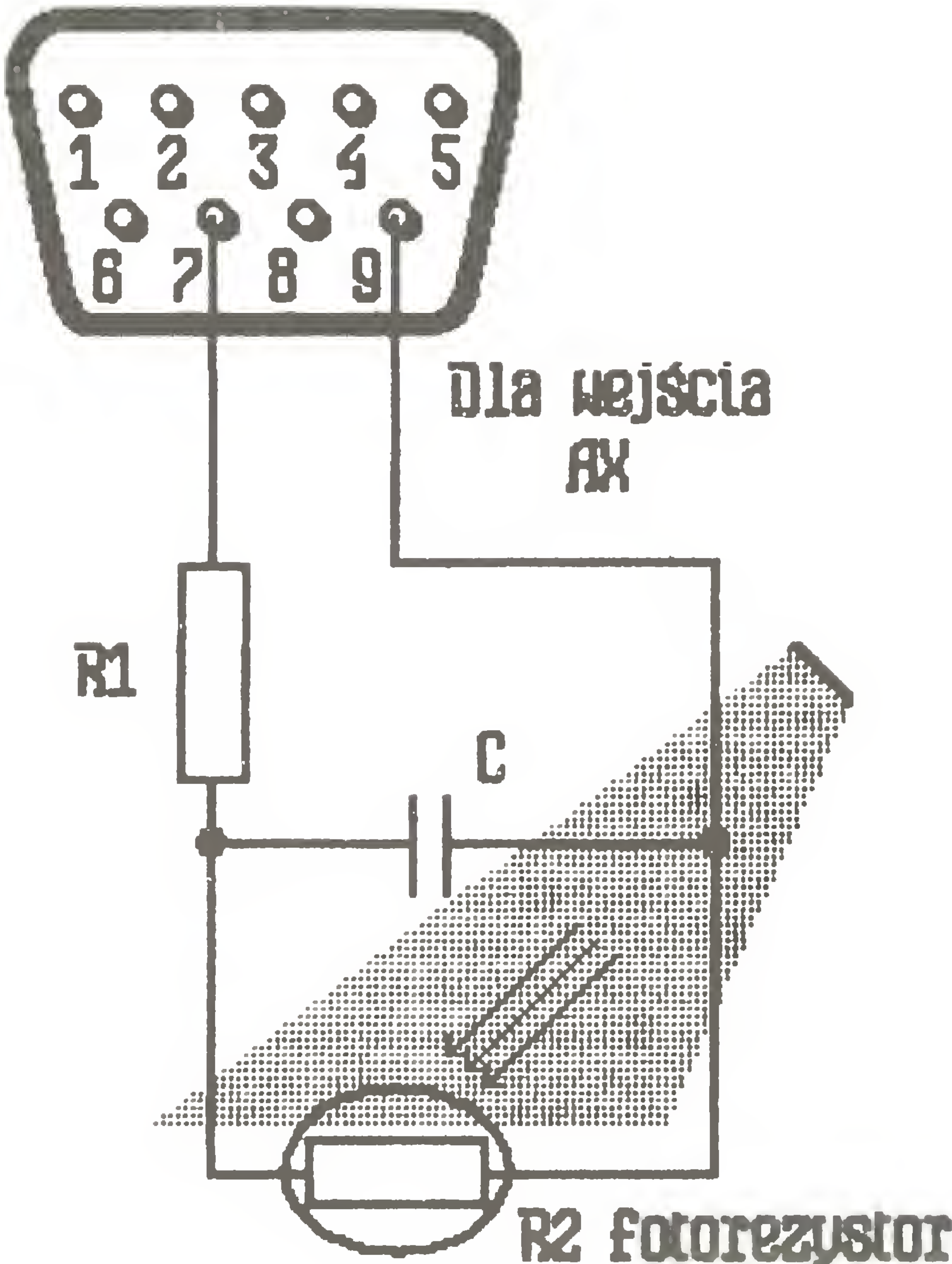
Dla drugiego portu funkcja odczytu jest identyczna, z tym że adres ładowany do rejestru BC jest o jeden większy (54298), pierwszy wiersz w procedurze Inline będzie następujący:

```
$01/$1A/$D4/ (* Ld bc,54298
             ($D41A) *)
```

Oczywiście dokładność uzyskiwana w pomiarach przy użyciu opisanych powyżej przetworników jest niewielka - zakres pomiarowy wynosi 0 - 255, a dodatkowo stosunkowo duża jest nieliniowość przetwarzania opisywanych przetworników. Wydaje się mimo to, że ta cecha C 64 i C 128 otwiera szerokie pole do eksperymentów.

Wszystkie podane powyżej informacje odnosić się mogą z równym powodzeniem do układu z fotoopornikiem jak i z termistorem (pomiar temperatury), lub dowolnym innym - dającym jako efekt wyjściowy zmianę oporności w zakresie 30 - 150k. Wykorzystując dwie opisane powyżej procedury można np. za pomocą dwóch czujników wykonywać co pewien czas pomiar temperatury lub innej wielkości fizycznej (natężenia światła, napięcia, prędkości wiatru itp.). Sposoby konstrukcji czujnika oraz to, co można mierzyć, pozostawiam inwencji Czytelnika.

Tak jak w powyższym przykładzie wykorzystaliśmy port przetwornika analogowo-cyfrowego, możemy wykorzystać wszystkie inne urządzenia wejścia-wyjścia C 128 - joysticki, User Port itd. ■



Jeżeli w bloku FCB występują znaki zapytania uaktywniany jest pierwszy element katalogu spełniający relację równości. Należy zauważyć, że bieżący rekord ("cr") musi być wyzerowany przez program jeżeli zbiór ma być dostępny sekwencyjnie od pierwszego rekordu.

```
*****************************************
*                                       *
* Funkcja 16: Zamknięcie zbioru         *
*            (CLOSE FILE)               *
*                                       *
*****************************************
*                                       *
* Parametry wejściowe:                  *
*         Rejestr    C:  10H            *
*         Rejestry DE: adres FCB        *
*                                       *
* Wartość   zwracana:                   *
*         Rejestr    A: kod katalogu    *
*                                       *
*****************************************
```

Funkcja "Zamknięcie zbioru" spełnia odwrotną rolę jak otwarcie zbioru zakładając, że blok FCB adresowany przez parę rejestrów DE, został wcześniej uaktywniony przez funkcję otwarcia lub utworzenia zbioru (patrz funkcje 15 i 22). Funkcja zamknięcia zbioru zawsze zapisuje nowy blok FCB w odpowiednim katalogu dyskowym. Proces dołączania bloku FCB dla zamknięcia jest analogiczny jak dla funkcji otwarcia. W przypadku zakończonej poprawnie operacji zamknięcia przekazywany jest kod katalogu równy 0,1,2 lub 3, natomiast gdy nazwa zbioru nie jest znaleziona w katalogu przekazywany jest kod 0FFH (255 dec). Zbiór nie wymaga zamknięcia, jeżeli ma miejsce tylko operacja czytania. Jeżeli natomiast miała miejsce operacja zapisu niezbędna jest operacja zamknięcia dla wpisania nowych danych do katalogu.

```
*****************************************
*                                       *
* Funkcja 17: Szukanie pierwszego       *
*            (SEARCH FOR FIRST)         *
*                                       *
*****************************************
*                                       *
* Parametry wejściowe:                  *
*         Rejestr    C:  11H            *
*         Rejestry DE: adres FCB        *
*                                       *
* Wartość   zwracana:                   *
*         Rejestr    A: kod katalogu    *
*                                       *
*****************************************
```

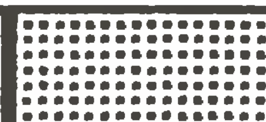
Funkcja powoduje poszukiwanie w katalogu pierwszego zbioru o nazwie odpowiadającej nazwie podanej w bloku FCB, który jest adresowany przez parę rejestrów

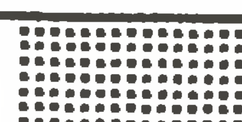
DE. Wartość OFFH (255 dziesiętnie) zostanie zwrócona, jeżeli zbiór nie zostanie znaleziony. W przeciwnym wypadku przekazywana jest wartość 0,1,2 lub 3 oznaczająca, że zbiór istnieje. Jeżeli zbiór zostanie znaleziony, wtedy do bufora (od bieżącego adresu DMA) zapisywany jest rekord zawierający dany element katalogu (adres względny tego elementu w buforze jest równy A*32). Dane katalogowe mogą być usunięte z bufora o tym adresie, jednak nie jest to wymagane dla programów użytkowych.

Znak zapytania 3FH (63 dziesiętnie) na dowolnej pozycji od "f1" do "ex", powoduje spełnienie relacji równości z odpowiadającym polem każdego elementu katalogu na standardowej lub wybranej przez użytkownika jednostce dyskowej. Jeżeli pole "dr" zawiera znak zapytania, funkcja wyboru dowolnej jednostki dyskowej jest zabroniona. Przeszukiwany jest dysk standardowy za pomocą funkcji szukania, która przekazuje każdy znaleziony element (przydzielony lub wolny) należący do dowolnego użytkownika. Ta ostatnia funkcja nie jest zwykle wykorzystywana przez program użytkowy, zapewnia ona jednak pełną elastyczność badania wszystkich bieżących wartości zawartych w katalogu. Jeżeli pole "dr" nie jest zapisane znakiem zapytania, bajt "s2" jest automatycznie zerowany.

```
*******************************************
*                                         *
* Funkcja 18: Szukanie następnego         *
*             (SEARCH FOR NEXT)           *
*                                         *
*******************************************
*                                         *
* Parametry wejściowe:                    *
*         Rejestr   C:  12H               *
*                                         *
* Wartość   zwracana:                     *
*         Rejestr   A: kod katalogu       *
*                                         *
*******************************************
```

Funkcja "Szukanie następnego" jest podobna do funkcji 17, z tą różnicą, że katalog jest przeszukiwany od ostatniego znalezionego elementu. Również, podobnie jak funkcja 17, funkcja 18 przekazuje w rejestrze A wartość OFFH (255 dec) jeżeli nazwa zbioru nie została znaleziona.


Mariusz Pietruszka




Łaty na Turbo Pascalu [1]






Nic nie jest tak doskonałe, aby nie dało się ulepszyć. Szczególnie wdzięcznym obiektem do wszelkich ulepszeń i poprawek jest znakomity kompilator - Turbo Pascal. Nasze poprawki dotyczyć będą tych wersji Turbo Pascala (2.0, 3.0..3.0A, 3.0E), które pracują na licznych w kraju 8- bitowych komputerach wyposażonych w system operacyjny CP/M- 80 (wersje 2.2 i 3.0).

Narzędzia

Aby móc praktycznie wykorzystać treść tego opracowania, potrzebne jest (i wystarczające) posiadanie następującego oprogramowania narzędziowego:

1. Rezydentne (CP/M 2.2) lub nierezydentne (CP/M 3.0) zlecenie SAVE.
2. Debuger (symboliczny) ZSID, lub inny: SID, DDT, DDTZ...

Zlecenie SAVE systemu CP/M 2.2 ma następującą składnię:

SAVE n nazwa

gdzie n jest liczbą 256-bajtowych stron, na które logicznie podzielona jest pamięć komputera pracującego pod kontrolą CP/M. Przyjmuje się, że początek obszaru, w którym znajduje się "debugowany" program, wynosi **100h** (jest to pierwszy adres obszaru TPA). Nazwa jest pełną (wraz z rozszerzeniem) nazwą dopuszczaną przez system. Jeśli zatem program (typu **COM**), w którym dokonujemy zmian, ma 4KB, dla n należy wpisać wartość 4*4=16=n, gdyż 1KB odpowiada czterem 256-bajtowym stronom. Zatem w wersji 2.2 CP/M wielkość obszaru pamięci, który chcemy zapisać na dysku, musi zostać obliczona przez operatora.

W przypadku wersji CP/M 3.0 należy wpisać tylko:

SAVE

Trzeba pamiętać, że nierezydujące zlecenie **SAVE** (czyli program **SAVE.COM**) wywołuje się przed uruchomieniem programu debugera. Program **SAVE** umieszcza się wtedy w pamięci operacyjnej komputera, gdzie czeka na wywołanie np. przez zlecenie **GO** (ang. *go*; adres = 0) podczas wyjścia z programu debugera. Wówczas program **SAVE** pyta o adresy pierwszego i ostatniego bajtu obszaru pamięci, który należy zapisać na dyskietce.

Metoda

Poniżej przedstawiono metodykę postępowania podczas zmian w programach CP/M.

1. Należy sprawdzić, czy na dyskietce znajduje się odpowiednia ilość miejsca do zapamiętania zmienionej przez nas wersji programu. Można to uczynić zleceniami **XDIR** lub STAT (CP/M 2.2) albo odpowiednio **SHOW** lub **DIR [SIZE]** (CP/M 3.0).

2. Wykonać zlecenie SAVE (tylko dla CP/M 3.0).

3. Wprowadzić z klawiatury zlecenie: **ZSID TURBO.COM.**

4. Odczytać i zapamiętać wartości szesnastkowo zapisanych adresów odpowiadających BEGIN i NEXT.

Korzystając ze zleceń debugera należy pamiętać, że każda pomyłka może "zaowocować" uszkodzeniem programu (czego efekt może ujawnić się dopiero po dłuższym czasie, np. po usunięciu poprzedniej wersji!).

Pracując z programem debugera będziemy używać podanych poniżej rozkazów symbolicznych.

La b (ang. *list*) - wypisz w mnemonikach procesora (deasembluj) instrukcje od **a** do **b**, gdzie **a** i **b** są adresami początku i końca - należy tu zwracać uwagę na składnię (spacje!).

Aa (ang. *assemble*) - asembluj program maszynowy od adresu a. ZSID zgłasza gotowość po ponownym wypisaniu określonego przez użytkownika adresu.

Sa (ang. *set*) - ustaw wartości pamięci. Pokazuje zawartość (w bajtach) komórki pamięci **a** i następnych. Po wypisaniu po lewej stronie ekranu adresu komórki pamięci i jej zawartości (bajtu), kursor przesuwa się nieco w prawo - program oczekuje wartości bajtu (hex.), który ma zostać wpisany pod ten adres. Wartości wpisane są zapamiętywane w pamięci RAM. Zakończenie po wprowadzeniu kropki.

Da b (ang. *display*) - wyświetl zawartość pamięci w bajtach (hex.) i odpowiadających im znakach ASCII, od adresu **a** do **b**.

Ga (ang. *go*) - rozpocznij wykonywanie programu od adresu **a**.

GO powoduje restart systemu (uruchamia także program **SAVE** - CP/M 3.0).

^C (ang. *Cancel*) - przerwanie pracy - powrót do systemu.

W przypadku CP/M 2.2 zlecenie rezydentne (czyli takie, jak np. **DIR, DIRS, TYPE) SAVE** o podanej wcześniej składni należy użyć bezpośrednio po dokonaniu zmian w poprawianym programie i po powrocie do systemu.

Przedstawiony wyżej opis obrazuje ogólny sposób postępowania. Natomiast bardzo istotna poprawka umożliwiająca użycie klawisza ESC była opisana w tekście "Sztuczki i chwyty [4]" w numerze 6/88 "Komputera". Poprawka ta dotyczyła często spotykanej, uszkodzonej wersji Turbo Pascala dla Amstrada CPC6128, 8256, 8512. Dla tych Czytelników, którzy nie mają pod ręką tego numeru naszego pisma przypominamy, że chodzi o zmianę wartości znajdującej się pod adresem 2E8Ch z 0FCh na 1Bh.

Oto symboliczny zapis całej operacji:

A> SAVE
A> ZSID TURBO.COM
NEXT BEGIN
7980h 100h
#S2E8C
#2E8C FC 1B
#2EBD .
#GO
(SAVE...)
A>

Powodzenia!

```
0745  D5        PUSH DE
0746  CD 0F 07  CALL 070FH
0749  F1        POP  AF
014A  EB        EX   DE,HL
074B  F0        RET  P
```

Rys. 2. Poprawka funkcji mod.

```
2C41  0E 1A      LD   C,1AH     ; Ustawienie adresu DMA
2C43  11 74 2C   LD   DE,2C74H  ; na 2C74H
2C46  CD 05 00   CALL 0005H
2C49  0E 19      LD   C,19H     ; Określenie aktualnego
2C4B  CD 05 00   CALL 0005H     ; napędu.
2C4E  0E 2E      LD   C,2EH     ; Określ. wolnego miejsca
2C50  5F         LD   E,A       ; (num. napędu do E)
2C51  CD 05 00   CALL 0005H     ; na dysku.
2C54  2A 74 2C   LD   HL,(2C74H); Uzyskana wartość do
2C57  3A 76 2C   LD   A,(2C76H) ; rejestrów HL i A.
2C5A  4F         LD   C,A       ; LSB do rejestru C.
2C5B  06 03      LD   B,3       ; Licznik pętli.
2C5D  AF   LOOP: XOR  A         ; Ustaw CY = 0.
2C5E  CB 19      RR   C         ; 3-bajtową liczbę podziel
2C60  CB 1C      RR   H         ; przez 2 (3 razy) -> otrzy-
2C62  CB 1D      RR   L         ; mamy wolne miejsce na
2C64  10 F7      DJNZ LOOP      ; na dysku w KB.
2C66  C3 77 2C   JP   2C77H     ; Skocz do "starej" proc.
2C91  DEFB 0,0,0,0              ; Wstaw 4 NOP-y
```

Rys.3. Poprawka odczytu wolnego miejsca na dysku pod systemem CP/M Plus.



Turbo Pascal – dziur w całym ciąg dalszy

Jacek Gwiazdka

Trzy opisane niżej poprawki dotyczą kompilatora Turbo Pascala w wersji 3.0 pracującej pod kontrolą systemu CP/M.

Pierwszy błąd jest bardzo poważny, został on wykryty i poprawiony przeze mnie. Błąd ten występuje przy konwersji najmniejszej liczby typu Integer, równej -32768, na postać zmiennoprzecinkową. W czasie wykonywania procedury dokonującej takiej konwersji program rozpoczyna nieskończoną pętlę, co powoduje zawieszenie się całego systemu. Przyczyną tego jest nieprawidłowe działanie zastosowanego algorytmu konwersji (ang. *integer → floating point)* dla liczb całkowitych, w których reprezentacji dwójkowej (w kodzie uzupełnienia do 2) występują same zera (nie biorąc pod uwagę bitu znaku). Liczba 0 jest już osobno rozpoznawana na początku procedury konwersji, aby więc procedura ta działała prawidłowo także dla liczby -32768 (w kodzie U2=8000H), powinna ona również osobno rozpoznawać ją na początku. Po rozpoznaniu liczby -32768 należy do par rejestrów HL,DE,BC załadować wartości odpowiadające zmiennoprzecinkowej reprezentacji tej liczby zastosowanej w Turbo Pascalu, czyli: wykładnik L=90H, najstarszy bajt mantysy B=80H i pozostałe bajty mantysy równe zeru (rejestry H,D,E,C). Rys. 1. przedstawia poprawkę dokonaną zgodnie z powyższymi uwagami. Poprawkę tę można umieścic na przykład w obrębie napisu firmowego Borlanda od adresu 0106H. Oczywiście należy również zmodyfikować początek "starej" procedury konwersji, umieszczając tam skok do nowego jej początku.

Z błędem tym związany jest następny, mniejszej wagi. Polega on na tym, że stałej -32768 nie można umieścić w tekście programu napisanego w Turbo Pascalu, ponieważ podczas kompilacji pojawia się komunikat o błędzie: "Error in Integer constans". Można to jednak zrobić korzystając z jej postaci szesnastkowej: $8000. Ten sam błąd występuje w Turbo Pascalu 3 pod kontrolą systemu MS-DOS [4], ale nie występuje już w wersji 4.0 pracującej pod kontrolą tego systemu.

Błąd drugi nie jest już tak "groźny" jak pierwszy, ponieważ nie powoduje zawieszania się systemu, a "tylko" niewłaściwe wyniki obliczeń. Błąd ten występuje w funkcji **mod** w przypadku ujemnego pierwszego argumentu.

Otrzymujemy: **-2 mod 5 = 2,** a powinno być: -2, i jednocześnie: **-8 mod 5 = -3** - czyli dobrze.

Rys. 2. przedstawia sposób poprawy tego błędu, dokonany na podstawie [2].

Trzeci błąd występuje tylko wówczas, jeśli kompilator Turbo Pascala wykorzystywany jest pod kontrolą systemu CP/M Plus. Istnienie tego błędu wynika z braku specjalnej wersji Turbo Pascala dostosowanej przez firmę Borland do systemu CP/M Plus.

> 16

```
0106  7C  ADRES:  LD  A,H       ; Czy liczba integer
0107  B5          OR  L         ; jest zerem?
0108  CA 72 0B    JP  Z,00B72H  ; Tak - zwróć liczbę FP = 0.
010B  FE 80       CP  80H       ; Czy jest to liczba
010D  C2 0D 10    JP  NZ,0100DH ; 8000h, czyli -32768?
0110  7D          LD  A,L       ; nie - skocz do procedury
0111  A7          AND A         ; konwersji Borlanda i kon-
0112  C2 0D 10    JP  NZ,0100DH ; tynuuj ją od: 100DH.
0115  CD 72 0B    CALL 00B72H   ; Tak wyzeruj rej: HL=DE=BC=0
0118  2E 90       LD  L,90H     ; Rejestry HL,DE,BC zawierają
011A  06 80       LD  B,80H     ; teraz liczbę -32768
011C  C9          RET           ; w postaci FP.


1008  C3 06 01    KONWERT: JP  ADRES
```

Rys. 1. Poprawka funkcji konwersji postaci liczby

```
program patchturbo;
{==========================================================
               1988 JACEK GWIZDKA, POLAND - ŁÓDŹ
==========================================================}
const
  sk1=3;  dlug=23;
  skok : array[1..sk1] of byte =
    ($C3,$06,$01);       {JP patch_adr ; tutaj patch_adr=0106H }
  patch_moj : array[1..dlug] of byte = (
    $7C,$B5,$CA,$72,$0B,$FE,$80,$C2,$0D,$10,$7D,A7,
    $C2,$0D,$10,$CD,$72,$0B,$2E,$90,$06,$80,$C9      );
  patch_adr = $0106; { Adres umieszczenia wstawki - Copyright
                                                   Borlanda}
  proc_adr = $1008; { Adres procedury Turbo Pascala
                      dokonującej konwersji liczb całkowitych na
                      zmiennoprzecinkowe, która zawierała błąd }
```

```
m1=7;
modpatch : array[1..m1] of byte = (
         $D5,$CD,$0F,$07,$F1,$EB,$F0);
modadr  = $0745;

  f1 = 40; f2 = 4;
  freeds_patch ; array[1..f1] of byte = (
     $0E,$1A,$11,$74,$2C,$CD,$05,$00,$0E,$19,$CD,$05,$00,
     $0E,$2E,$5F,$CD,$05,$00,$2A,$74,$2C,$3A,$76,$2C,$4F,
     $06,$03,$AF,$CB,$19,$CB,$1C,$CB,$1D,$10,$F7,$C3,$77,$2C
  );
  fzeruj_patch ; array[1..f2] of byte = (0,0,0,0);
  freeds_adr = $2C41;
  zeruj_adr  = $2C91;

var
  tx : file;
  i : integer;

begin
  { Patch 1: konwersja -32768 na FP. }
  move(skok,mem[proc_adr,sk1);
  move(patch_moj,mem[patch_adr],dlug);

  { Patch 2 : Funkcja mod. }
  move(modpatch,mem[modadr],m1);


  { Patch 3 : Wolne miejsce na dysku }
  move(freeds_patch, mem[freeds_adr],f1);
  move(zeruj_patch,mem[zeruj_adr],f2);
  assign(tx,'TURBOJG.COM');
  rewrite(tx);
  blockwrite(tx,mem[$0100],242);
  close(tx);
end.
```

Rys. 4. Program automatycznej modyfikacji kompilatora TURBO Pascal (zbiór wyjściowy: TURBOJG.COM)

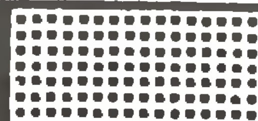
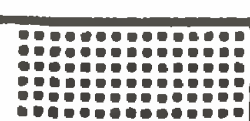
15 <

Przy pracy z Turbo Pascalem pod kontrolą systemu CP/M Plus podaje on nieprawidłowo wolne miejsce na dyskietce (komenda Dir menu głównego). Wynika to z wykorzystywania przez ten kompilator wywołań funkcji BIOS, które są w nieco inny sposób zrealizowane w CP/M Plus niż w CP/M 2.2. Rys. 3. zawiera, na podstawie [3], poprawki niezbędne do usunięcia tego błędu.

Rys. 4. przedstawia program napisany w Turbo Pascalu, dokonujący automatycznie trzech podanych wcześniej poprawek.

Literatura:
1. Turbo Pascal version 3.0 Reference Manual, Borland Inc.;
2. "Neues aus Turbanien", c't Magazin 10/1986;
3. F. Renz, "Phantom gefangen", c't Magazin 8/1986;
4. Andrzej Kadlof "Dziury w całym", "Komputer" 3/1988.

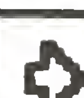


  **W pracy**  

Tomasz Mazur

Mikroprogramy dla Atari XE/XL

W poprzednim odcinku "Mikroprogramu" zajęliśmy się dość prostymi programami.Dziś nadeszła pora na bardziej skomplikowane. Autorem pierwszych trzech jest Leszek Taratuta z Olsztyna. Tylko pozornie są to programy demonstracyjne. Leszek w swoich programach wykorzystuje przerwania listy displejowej, a utworzone procedury pozwalają zmieniać kolory w pojedynczych liniach skaningowych.

Zacznijmy od najprostszego z tych programów, pozwala on na uzyskanie 128 kolorów w trybie GRAPHICS 0. Wszystkie trzy programy napisane zostały w języku maszynowym, a Basic służy jedynie do załadowania na 6. stronę pamięci procedury zapisanej w instrukcji DATA. Tekst źródłowy w asemblerze podamy tylko w przypadku pierwszego, najprostszego programu. Gdyby ktoś z Państwa chciał "rozszyfrować" pozostałe, wystarczy zamienić liczby znajdujące się w instrukcji DATA na odpowiednie rozkazy mikroprocesora i wartości liczbowe (odpowiedniki można znaleźć w "ATARI BASIC" pod redakcją Wiesława Miguta). W każdym z prezentowanych programów przed wpisaniem procedury maszynowej należy wyłączyć 7. bit w rozkazie ANTIC-u (informacja o przerwaniu DLI - Display List Interrupt), zapisać w komórkach 512 i 513 adres, gdzie zapisana jest procedura DLI. A poprzez POKE 54286,192 "przekonać" komputer o zezwoleniu na wykonanie przerwania DLI.

Drugi z programów pozwala uzyskać na ekranie monitora dobrze znaną atarowską tęczę, czyli poziome, kolorowe pasy sprawiające wrażenie ruchu.

Wiersze 10, 20 to oczywiście załadowanie procedury na 6. stronę pamięci czyli od adresu 1536, w wierszu 30. wspomniane już dane wstępne: wyłączenie 7. bitu ANTIC-u, podanie adresu procedury i zezwolenie na przerwanie DLI. Wiersze 100-120 to procedura programu zapisana przy użyciu kodów cyfrowych rozkazów mikroprocesora. Aby uzyskać program źródłowy, należy sprawdzić odpowiednie wartości w tabeli kodów dziesiętnych i mnemoników (pamiętając oczywiście czy rozkaz jest jedno-, dwu- czy trzybajtowy), np.:

Tekst źródłowy w asemblerze do programu 128

```
      PHA
      TXA
      PHA
      LDX #0
SKOK  STA 54282    ;WSYNC
      STX 53272    ;COLPF2
      INX
      INX
      BNE SKOK
      PLA
      TAX
      PLA
      RTI
```

Program 128

```
10 FOR A=1536 TO 1554
20 READ X: POKE A,X: NEXT A
30 POKE 39980,130:POKE 512,0:POKE 513,6:POKE 54286,192
40 DATA 72,138,72,162,0,142,10,212,142,24,208,232
50 DATA 232,208,246,104,170,104,64
```

72 → PHA – jednobajtowy, zapis zawartości akumulatora na stos;

138 → TXA - jednobajtowy, zawartość rejestru X do akumulatora;

72 → PHA - jak wyżej;

152 → TYA - jednobajtowy, zawartość rejestru Y do akumulatora;

72 → PHA - jak wyżej;

166, 200 → LDX C8 - dwubajtowy, wpisuje zawartość komórki 200 (heksadecymalnie C8) do rejestru X;

itd.

Program FALA

```
 10 FOR I=1536 TO 1562
 20 READ X: POKE I,X: NEXT I
 30 POKE 39970,240:POKE 512,0:POKE 513,6:POKE 54286,192
100 DATA 72,138,72,152,72,166,200,160,192
110 DATA 141,10,212,142,24,208,232,136,208,246,230,200
120 DATA 104,168,104,170,104,64
```

Trzeci z przysłanych przez Leszka programów pozwala na uzyskanie na ekranie statycznego obrazu tęczy. W trybie GRAPHICS 2 cieniowane jest zarówno tło jak i litery.

Program TĘCZA

```
 5 GRAPHICS 2+16
 10 FOR A=1536 TO 1610
 20 READ X: POKE A,X: NEXT A
 30 DL=PEEK(560)+256*PEEK(561)
 40 POKE DL+1,112+128:POKE 512,0:POKE 513,6:POKE 54286,192
 50 FOR A=0 TO 9 STEP 3: POSITION 5,A:? #6;"ATARI 65 XE ":
POSITION 5,A+1:? #6;"GRAPHIC II":NEXT A
 60 GOTO 60
 70 DATA 72,138,72,152,72,162,14,160,0,169,12,133,201,169,8
 75 DATA 133,200,141,10,212,140,26,208,142,22,208,200,200,202
 80 DATA 202,198,200,208,239,169,8,133,200,136,136,232,232
 85 DATA 141,10,212,140,26,208,142,22,208,198,200,208,239
 90 DATA 152,24,105,16,168,138,24,105,16,170,198,201,208
 95 DATA 200,104,168,104,170,104,64
```

Oczywiście wiersz 50 można dowolnie modyfikować. Wykorzystana w programie procedura zmieniająca rejestry kolorów COLBAK COLPFO jest bardzo prosta i składa się z dwóch części. Pierwsza zwiększa zawartość COLBAK, a zmniejsza COLPFO, zaś druga odwrotnie, w ten sposób uzyskuje się efekt "cieniowania" kolorów. Program powyższy możemy także modyfikować, np. zmieniając w wierszu 40 POKE DL+1,112+128 na POKE DL+2,112+128, otrzymamy odwrotnie ułożone barwy.

Możemy także wprowadzić zmiany w komórkach 1944 i 1942. Pierwsza z nich odpowiada za rejestr startowy COLBAK - normalnie jest tam 0 (kolor czarny), jeśli zaś wpiszemy wielokrotności 16, możemy uzyskać inne kolory tła i znaków w jednym wierszu. Analogicznie postępujemy dla COLPFO, tyle że zmieniamy zawartość komórki 1942 (normalnie jest 14 - kolor biały) wpisując wartości zgodnie ze wzorem 16*n +14, gdzie n jest kodem koloru. Mogą także Państwo spróbować połączyć tę procedurę z efektem falowania uzyskanym w poprzednim programie. Tylko że taki program będzie na pewno dłuższy niż 15 wierszy.

Na zakończenie niespodzianka od Pawła Koziela z Limanowej dla wszystkich użytkowników magnetofonów - program testujący prędkość i nierównomierność przesuwu taśmy w magnetofonie. Wykorzystano w nim procedurę SIO, która dokonuje korekcji prędkości transmisji przed wczytaniem każdego rekordu danych. Po uruchomieniu programu należy dokonać zapisu na taśmie wzorcowej. Najkorzystniej będzie, jeżeli do tego celu użyjemy dobrego, sprawdzonego magnetofonu o małej nierównomierności przesuwu taśmy (do ok. 0,15%) i prędkości średniej równej nominalnej (4,76 cm/s). Jeżeli posiadamy już taśmę wzorcową, możemy przystąpić do pomiaru (używając opcji 2). Podczas wczytywania programu wzorcowego na ekranie pojawi się wykres obrazujący zmianę prędkości i chwilowa jej wartość. Po odczytaniu całego zapisu wzorcowego program poda średnią prędkość przesuwu i odchylenie tej wartości od nominalnej w procentach. Natomiast na wykresie pojawią się trzy linie poziome. Środkowa obrazuje średnią prędkość przesuwu taśmy, zaś skrajne wyznaczają pole tolerancji dla nierównomierności mniejszej lub równej 0,5%.

W wierszach 10-30 tworzone jest menu programu, wiersze 40 i 50 to zapis taśmy wzorcowej. W wierszach od 60 do 100 zawarty jest główny blok programu, to znaczy: odczyt taśmy wzorcowej, obliczanie wartości bieżącej i średniej prędkości, rysowanie wykresu i linii prędkości średniej. Wiersze 110-150 to obliczanie i rysowanie pola tolerancji i "kosmetyka" wyników programu.

Program TEST

```
 10 OPEN #1,4,0,"K:"
 20 GRAPHICS 0:POKE 752,1:? " TEST MAGNETOFONU ":? :
? "1.ZAPIS":? "2.POMIAR"
 30 TRAP 20:GET #1,B:ON B-48 GOTO 40,60
 40 ? CHR$(125):? "ZAPIS":OPEN #2,8,0,"C:"
 50 FOR I=1 TO 50:POSITION 8,1:? 51-I,:POKE 61,127:PUT #2,1:
NEXT I:RUN
 60 GRAPHICS 8:COLOR 1:POKE 752,1:? "POMIAR":
OPEN #2,4,0,"C:":X=6:S=0:PLOT 0,84
 70 POKE 656,0:? "Prędkość aktualna    cm/s;FOR I=1 TO 50:
GET #2,B:POKE 61,128
 80 V=((PEEK(750)+256*PEEK(751))/1480)*4.76:GOSUB 150
 90 DRAWTO X,84+400*(4.76-V):X=X+6:S=S+V:POKE 656,0:
POKE 657,20:? V;" "
100 NEXT I:V=S/50:GOSUB 140:? CHR$(125):
? "Prędkość średnia ";V;" cm/s ";
110 P=V:V=(V-4.76)*100/4.76:GOSUB 150:? "(";V;"%) "
120 V=1.005*P:GOSUB 150:GOSUB 140:V=0.995*P:GOSUB 150:
GOSUB 140
130 ? " [SPACE] menu ";:GET #1,B:RUN
140 Y=84+400*(4.76-V):PLOT 0,Y:DRAWTO 315,Y:RETURN
150 V=V*100:V=(INT(V)+((V-INT(V))>0.5))/100:RETU...
```

Mam nadzieję, że procedury zawarte w powyższych programach okażą się Państwu przydatne. Jednocześnie chciałbym przypomnieć, że oczekujemy na listy nie tylko z programami, ale także z ocenami zamieszczanych programów, aby wybrać ten, który się Państwu najbardziej podobał.

Panu Andrzejowi Grzesiakowi z Warszawy dziękuję za uwagi i prawdziwy mikroprogram realizujący kasowanie wybranych wierszy programu (patrz "Komputer" 2/88). W programie A- numer pierwszego wiersza do skasowania, B-numer ostatniego.

```
0 INPUT A,B:? CHR$(125):FOR
I=A TO B:P
OSITION 2,4:? I:? "CONT":POKE
842,13:P
OSITION 2,0:? " ":STOP
1 POKE 842,12:?
CHR$(125):NEXT I=LIST
```

Chciałbym podziękować także za list 12-letniemu Sebastianowi z Cieszyna, który swoim prostym programem do rysowania joystickiem udowodnił, że także młodzi użytkownicy komputerów zajmują się nie tylko grami.

\W domu\

Artur P. Chmielewski

Kompletny katalog dysku

W związku z rosnącym zainteresowaniem komputerami Atari ST pojawia się problem jak najefektywniejszego ich wykorzystania. Jedni poszukują coraz to nowych programów tworzonych przez wyspecjalizowane firmy, inni starają się, w miarę swoich możliwości, pisać programy przeznaczone do użytku prywatnego. Nie zawsze jednak konieczne jest pisanie takiego programu zupełnie od początku. Istnieje bardzo wiele gotowych procedur przeznaczonych do wykorzystywania przez programistów-amatorów w ich pracy. Jedną z takich procedur chciałbym właśnie przedstawić.

Na listingu 1 znajduje się tekst źródłowy programu drukującego na standardowym wyjściu komputera (czyli najczęściej na ekranie monitora) spis wszystkich plików zapisanych na dysku umieszczonym w aktywnej stacji dysków. Otwiera on sam i przegląda wszystkie katalogi, rozpoczynając od głównego katalogu dyskietki.

Program napisany jest w języku C i skompilowany za pomocą kompilatora **Mark Williams C Compiler Vers.2.0**. Składa się z kilku podstawowych części: pierwsza (wiersze 1-3), to dyrektywy preprocesora C: "include", które powodują dołączenie do tekstu źródłowego jeszcze przed rozpoczęciem właściwej kompilacji kilku niezbędnych definicji stałych i struktur danych oraz makrodefinicji funkcji (np. *Dsetpath(s)* - makro rozwijane do postaci *gemdos(59,s)*).

Następną częścią programu (wiersze 5 - 15) jest definicja funkcji *czas()*, zwracającej wartość typu (char *). Funkcja ta zamienia datę utworzenia lub ostatniej modyfikacji pliku z postaci mapy bitowej w danej typu *(unsigned long)* do postaci czytelnej dla człowieka - łańcucha ASCII (format: rr.mm.dd gg:mm).

Kolejna (wiersze 17 - 41) część programu - to definicja rekursyjnej funkcji *dir()*, przeglądającej wszystkie katalogi dysku w poszukiwaniu zapisanych tam plików. Wymaganym argumentem wywołania jest ścieżka, od której zaczyna się przeglądanie dyskietki. Nazwa ścieżki powinna być podana bez kończącego znaku "back-

> 18

17 <

slash" (w *main()* wywołanie *dir("")*; nakazuje więc zaczęcie przeglądania od katalogu głównego - ang. *root*).

Czwarty i ostatni fragment (wiersze 43 - 49) - to definicja *main()* - funkcji wykonywanej zawsze jako pierwszej w programie napisanym w C. Na początku następuje wypisanie na ekranie czołówki programu, następnie ustawienie bieżącej ścieżki dostępu do katalogu głównego aktywnego dysku i w końcu wywołanie *dir()*, wypisującej gotowy spis plików. Pliki czytane są w takiej kolejności, w jakiej znajdują się w fizycznym katalogu dysku (nie jest to więc np. kolejność alfabetyczna!).

Nie będę dokładnie opisywał działania programu i samego algorytmu, zostawiam to dociekliwości Czytelników. Chciałbym tylko skoncentrować się na bardziej szczegółowym przedstawieniu kilku procedur systemowych (GEMDOS) oraz podsunąć parę wskazówek przydatnych przy przystosowywaniu funkcji *czas()* i *dir()* dla potrzeb własnych programów.

Najpierw może jednak o tym, jak dojść do otrzymania gotowego programu **fdir.prg**. Po uruchomieniu systemu **Mark Williams C Compiler** ("Komputer" 2-3/88) wywołujemy edytor ekranowy MicroEMACS komendą **"me fdir.c"** i przepisujemy tekst źródłowy programu z listingu 1 (UWAGA: nie trzeba przepisywać numerów wierszy) pamiętając o tym, by znak *hash* (#) rozpoczynający dyrektywę "include" znajdował się w pierwszej kolumnie tekstu. Następnie zapisujemy plik i "wychodzimy" z edytora (CTRL-Z), po czym kompilujemy program zleceniem **"cc fdir.c"**. Jeśli nie popełniliśmy w trakcie przepisywania żadnych błędów, po chwili na dysku mamy gotowy zbiór **"fdir.prg"**, który możemy (i oczywiście robimy to) uruchomić wpisując **"fdir"**. Na ekranie otrzymujemy przykładowo następujący tekst:

```
a:\> fdir
FULL DISC Directory Vers.3.0 (DEMO)
by (C) KAY SOFTWARE '88
Artur P. Chmielewski
FDIR.PRG          7886 00 88.03.17 22.23
FDIR.C            1352 00 88.03.17 22.14
FORMAT.PRG        5230 00 86.02.06 00:04 \FORMAT
FORMAT:RSC        1236 00 86.02.06 00:04 \FORMAT
MINICAD.PRG      21500 00 86.02.06 00:04 \MINICAD
MINICAD.RSC       6194 00 86.02.06 00:04 \MINICAD
WATCH.ACC         5441 20 86.02.06 00:05
RAMDISK.TOS       1289 00 86.02.06 00:05
HISCORE              6 00 86.02.06 00:22 \MEGAROID.S
MEGAROID.PRG     46720 00 86.02.06 00:22 \MEGAROID.S
MEGAROID.RSC      2560 00 86.02.06 00:23 \MEGAROID.S
a:\>
a:\>ls -l ?a*.*
-rw— 1289 Thu Feb 6 1986 ramdisk.tos
-rw–m 5441 Thu Feb 6 1986 watch.acc
a:\>
```

Przed przejściem do zasadniczego tematu jeszcze parę słów wyjaśnienia: w dalszych partiach tekstu napotkać można angielskie wyrażenie "Wild Cards". Z braku jego polskiego odpowiednika (najlepiej chyba oddaje istotę rzeczy określenie meta-znaki) posługuję się nim w oryginalnym brzmieniu. Chciałbym tylko wyjaśnić jego znaczenie: Wild Cards to specjalne znaki zastępujące np. w nazwie pliku inne znaki. Najczęściej spotyka się '?' zamiast dowolnego pojedynczego znaku oraz '*' zamiast dowolnej grupy znaków. Dla przykładu z nazw "kay.doc", "lala.doc", "out.c" i "malarz.rot" kryteriom określonym przez "?a*.?o?" odpowiadają wszystkie nazwy poza "out.c".

Zajmijmy się teraz omówieniem występujących w programie wywołań systemowych. Są to (w nawiasach znajdują się numery odpowiedniej funkcji GEMDOS-a w postaci szesnastkowej):

Dsetpath (0x3B), Fgetdta (0x2F), Fsetdta (0x1A),
Fsfirst (0x4E), Fsnext (0x4F), Cconws (0x09).

Dsetpath wymaga jednego argumentu: wskaźnika do początku łańcucha tekstowego zawierającego nazwę ścieżki dostępu (ang. *Path*) ustawianej jako aktywna. Podejrzewam, iż pojęcie "ścieżka dostępu" znane jest lepiej osobom, które miały lub mają do czynienia również z innymi poza DESKTOP-em procesorami komend (ang. *Consol Command Processor*), np. z powłoką UNIX-a (aktywna ścieżka wyświetlana zleceniem **pwd** - *print working directory)* lub chociażby z CCP MS DOS czy CP/M. W DESKTOP-ie nazwa bieżącego katalogu znajduje się w tytule aktywnego okna.

Dla przykładu **Dsetpath("a:/mwc/src");** ustawia jako aktywny katalog **mwc/src** na dysku A:. W przypadku operacji zakończonej sukcesem **Dsetpath** zwraca wartość 0, w przeciwnym razie liczbę różną od zera.

Fgetdta zwraca wartość typu (DMABUFFER *) czyli wskaźnik do 44-bajtowej struktury zadeklarowanej w pliku **stat.h** w następujący sposób:

```
typedef struct {
  char d_glob[12];  /* GEMDOS wildcard lub nazwa z Fsfirst */
  char d_mask;      /* Maskowanie atrybutów zbioru z Fsfirst */
  char d_dirent[4]; /* Offset katalogu, pierwsze MSB */
  char d_dirid[4];  /* Wskaźnik do struktury opisującej katalog */
  char d_fattr;     /* Atrybuty zbioru */
  long d_tandd;     /* Czas i data */
  long d_fsize;     /* Długość zbioru */
  char d_fname[14]; /* Nazwa zbioru */
}DMABUFFER;
```

Struktura ta używana jest przez funkcje **Fsfirst()** oraz **Fsnext()**, omówione w dalszej kolejności, a zawiera kompletne dane dotyczące danego pliku odczytanego przez **Fsfirst** lub **Fsnext**.

Funkcją o działaniu przeciwnym do **Fgetdta** jest **Fsetdta**. Wymaga ona jednego parametru - wskaźnika do struktury DMABUFFER i nie zwraca żadnej wartości (jest to więc funkcja typu *void*). Ustawia ona adres bufora DMA (struktura DMABUFFER) na adres wskazany przez argument swego wywołania.

Funkcja (a właściwie, tak jak i wszystkie poprzednie - "makro" zdefiniowane w pliku osbind.h) **Fsfirst** wymaga dwóch argumentów: wskaźnika do nazwy poszukiwanego pliku, (najczęściej w Wild Cards) oraz liczby typu (int) określającej jego atrybuty. Drugi parametr może przyjmować następujące wartości (szesnastkowo):

0x00	**-tylko normalne pliki**
0x01	**-również pliki tylko do czytania**
0x02	**-również pliki ukryte**
0x04	**-również pliki systemowe**
0x08	**-również etykiety dysku**
0x10	**-również katalogi i podkatalogi.**

Fsfirst poszukuje pierwszego wystąpienia pliku spełniającego podane warunki w bieżącym katalogu i w wypadku znalezienia takowego zwraca wartość **AE_OK** (stała zadeklarowana w pliku **errno.h** i równa 0L) oraz wypełnia 44- bajtowy bufor założony przez **Fsetdta** informacjami o tym pliku (do zawartości bufora najlepiej odwoływać się przez opisaną powyżej strukturę DMABUFFER). W przypadku nieznalezienia odpowiedniego pliku **Fsfirst** zwraca wartość różną od zera (najczęściej **AEFILNF** - 33L - *file not found).*

Funkcją używaną zwykle razem z **Fsfirst** jest **Fsnext**. Szuka ona następnego wystąpienia pliku na podstawie informacji zapisanej w buforze DMA przez **Fsfirst** lub poprzednie wywołanie **Fsnext**. W wypadku znalezienia następnego pliku funkcja uaktualnia zawartość struktury DMABUFFER danymi go dotyczącymi i zwraca **AE_OK**.

Ostatnim wywołaniem systemowym GEMDOS-a jest w tym programie **Cconws**. Funkcja ta wymaga jednego argumentu - wskaźnika do tekstu (zakończonego oczywiście - jak wymaga tego standard C - znakiem NUL - '\0'), który to tekst zostaje wypisany na standardowym wyjściu komputera, czyli najczęściej na ekranie monitora (urządzenie logiczne **con:**). Oczywiście posługując się np. **MicroShellem** możemy "przekierunkować" wyjście programu i zamiast na ekran wysłać je do pliku **"out.dir"**:

```
a:\> fdir >out.dir
a:\> cat out.dir
FULL DISC Directory Vers.3.0 (DEMO)
by (C) KAY SOFTWARE '88
Artur P. Chmielewski

FDIR.PRG          7886 00 88.03.17 22.23
FDIR.C            1352 00 88.03.17 22.14
FORMAT.PRG        5230 00 86.02.06 00:04 \FORMAT
FORMAT.RSC        1236 00 86.02.06 00:04 \FORMAT
WATCH.ACC         5441 20 86.02.06 00:05
RAMDISK.TOS       1289 00 86.02.06 00:05
MEGAROID.PRG     46720 00 86.02.06 00:22 \MEGAROID.S
```

MEGAROID.RSC 2560 00 86.02.06 00:23 \ MEGAROID.S
a:\ >
lub na drukarkę:
a:\ > fdir >prn:

Przedstawiony program jest bardzo skromny. Można go w prosty sposób rozbudować dodając np. rozpoznawanie etykiet dysku, gromadzenie danych dotyczących plików i poddawanie ich dalszej obróbce (sortowaniu, wybieraniu niektórych nazw itd.), wyposażyć go w wygodne menu (np. okna, sterowanie myszką, ikony) lub wbudować w jakiś większy pakiet np. edytora dyskowego.

Przy okazji chciałbym poinformować Czytelników, iż taki program już istnieje i nazywa się **ST_F.DIR.** Programik wydrukowany poniżej jest tylko jego drobnym fragmentem (ale dość istotnym). Całość dostępna jest w klubie ST (przy redakcji "Komputera") jako program *Public Domain.* **ST_F. DIR** pracuje praktycznie na każdej konfiguracji Atari ST (monitor monochromatyczny oraz kolorowy w trybie rozdzielczości *Medium*), ze wszystkimi rodzajami stacji dysków oraz HARD - i RAM-dyskiem (oznaczenia napędów od A do P), sortuje pliki wg nazw, długości, dat zapisu, typów *(extender)* lub ścieżek dostępu, pozwala na wybór dowolnego formatu wyświetlania danych, drukuje przez złącze RS 232 C, Centronics lub zapisuje listę plików na dysku w postaci zbioru ASCII, ma możliwość wypisywania drzewa katalogów dysku.



```
#include <osbind.h>        /* makra gemdos, xbios i bios        */
#include <stat.h>          /* struktura DMABUFFER, itp.         */
#include <time.h>          /* struktury utetd_t, rtetd_t, itp.  */

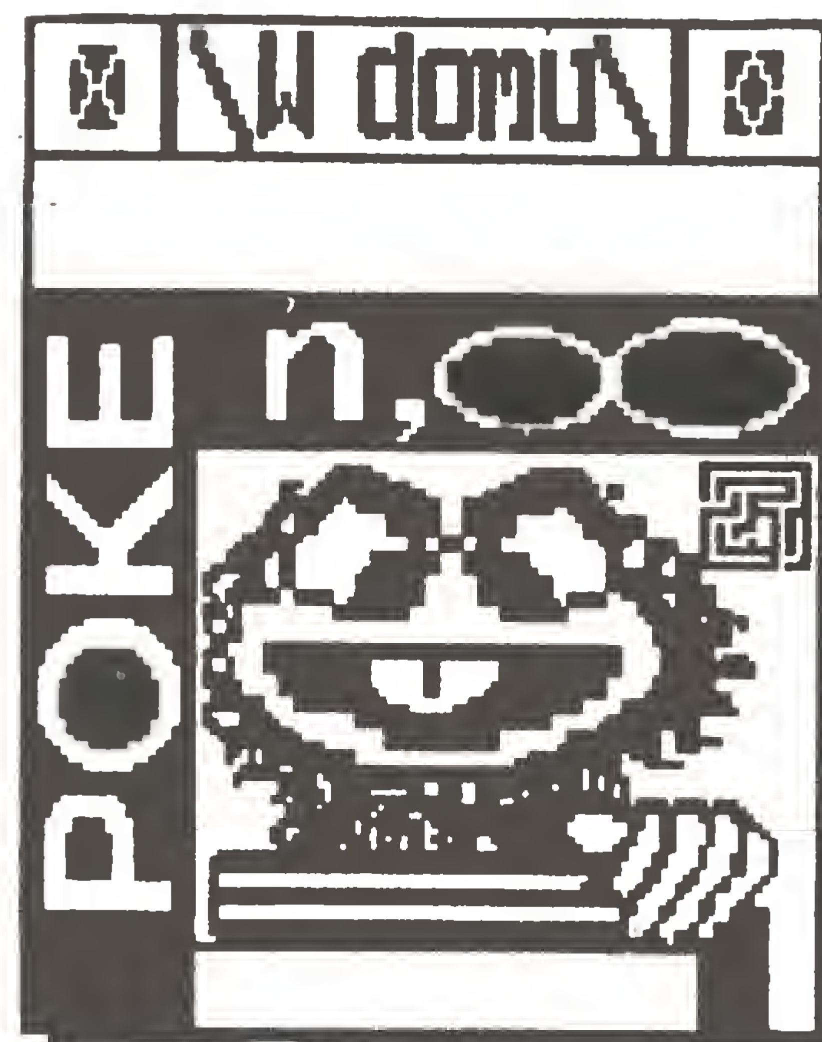
char *czas(n)
unsigned long n;
{  utetd_t utd; rtetd_t rtd; tm_t *tp;
   static char *tab[20];

   *((long *)&rtd) = n; utd.g_date = rtd.g_rdate; utd.g_time = rtd.g_rtime;
   tp = tetd_to_tm(utd);
   sprintf(tab,"%2d.%02d.%02d %02d:%02d",tp->tm_year, tp->tm_mon + 1,
        tp->tm_mday, tp->tm_hour, tp->tm_min);
   return (char *)tab;
}

dir(s)
char *s;
{  DMABUFFER *saved, ndta;
   char ptr[70], buf[70];

   ptr[0] = 0;
   if (Dsetpath(s) != 0) return;
   saved = (DMABUFFER *)Fgetdta();
   Fsetdta(&ndta);                              /* nowy bufor dla DMA */
   if (Fsfirst("*.*",0xff) == 0) {
      do {
         if (ndta.d_fattr == 0x10) {            /* gdy plik to katalog */
            if (ndta.d_fname[0] != '.') {
               sprintf(ptr,"%s\\%s",s,ndta.d_fname);
               dir(ptr);            /* tu nastepuje wywolanie rekursyjne */
            }
         } else {                               /* gdy nie jest to katalog */
            sprintf(buf,"%-14s %71d %02x %s %s\n\r",ndta.d_fname,
               ndta.d_fsize,ndta.d_fattr,czas(ndta.d_tandd),s);
            Cconws(buf);                 /* wyswietlenie kolejnego pliku */
         }
      } while (Fsnext() == 0);                  /* dopoki sa jakies pliki */
   }
    Fsetdta(saved);                            /* stara wartosc bufora DMA */
}

main()
{
   Cconws("FULL DISC Directory Vers.3.0 (DEMO));
   Cconws(\n\rby (C) KAY SOFTWARE '88\n\rArtur P. Chmielewski\n\r\n\r");
   Dsetpath("\\");                         /* katalog root aktywnego dysku */
   dir("");      /* wywolanie dir() dla katalogu biezacego czyli root */
}
```



AMSTRAD CPC 6128

W **ANTIRIAD** (trzeci program firmy Palace Software na rynku Amstrada po obydwu grach CAULDRON) zupełną odporność na kolizje z obcymi przedmiotami uzyskamy po modyfikacji czwartego pliku programu o nazwie ANTIRIAD.N03 (adres umieszczenia &3000, długość &52000).
20 a$="ANTIRIAD.N03":
ad=&3000:le=&52000:ex=0:
sum=12390
130 DATA 33,0,195,34,190,55,201

Program **AVENGER** był już omawiany, proponowane poprawki zapewniają "wieczne życie", uniwersalny klucz oraz zwiększają zapasy broni.
20 a$="AVENGER1.BIN":
ad=&200:le=&A380:ex=0:
sum=14522
130 DATA 175,50,142,162,50,143,
162,50,144,162,62,255
140 DATA 50,211,106,62,201,50,
239,163,201

Gry **THREE WEEKS IN PARADISE, BOULDER DASH, EQUINOX, PAPER BOY** oraz **STARQUAKE** również pojawiały się na łamach "Komputera". Modyfikacje wprowadzone do tych programów zapewnią nam "wieczne życie", w BOULDER DASH zatrzymają dodatkowo upływ czasu, a w PAPER BOY dadzą nam jeszcze nieograniczony zapas gazet.

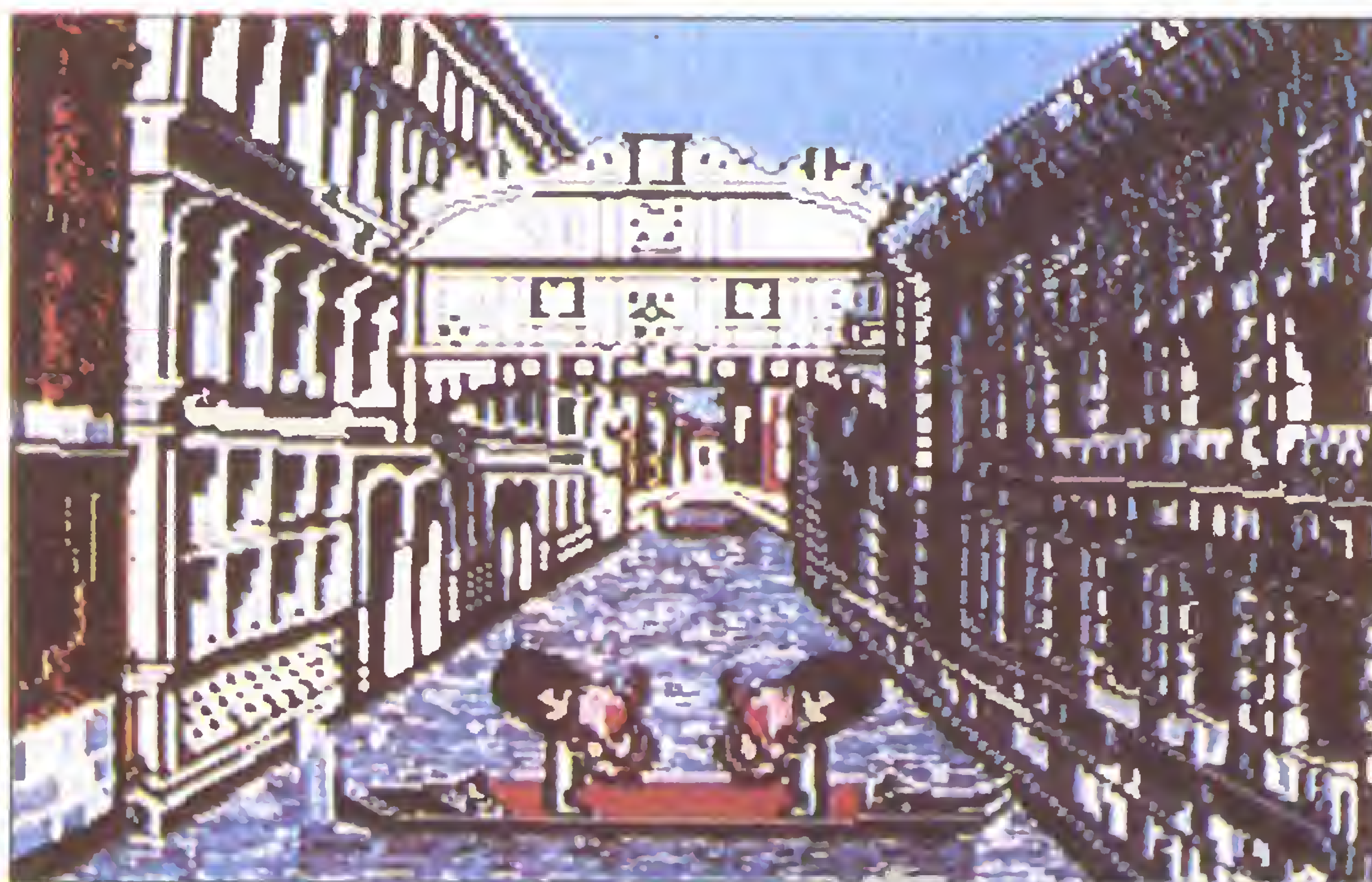
● **THREE WEEKS IN PARADISE:**
20 a$="PARADISE.PRG":
ad=&100:le=&9F01:ex=103:
sum=12371
130 DATA 175,50,206,57,201

● **BOULDER DASH:**
20 a$="DASH1.BIN":ad=&200:
le=&7036:ex=1F52:sum=13498
130 DATA 175,50,215,8,50,216,8,50,
217,8,50,128,27,50
140 DATA 129,27,50,130,27,201

● **EQUINOX:**
20 a$="EQUINOX1.BIN":ad=&40:
le=&A63C:ex=0:sum=12186
130 DATA 62,58,50,130,3,201

● **PAPERBOY:**
20 a$="PAPERBOY.N03":
ad=&40:le=&8F00:ex=0:
sum=12470
130 DATA 175,50,71,5,50,231,5,201

W programie **STARQUAKE** kody teleportacyjne różnią się trochę od wersji ZX Spectrum i są następujące: VOREX, TALIS, DULON, ANGLE, KRYZL, INDOL, RALIQ, ASCIO, QUORE, ELIXA, UPAZZ, OPTIK, SNODY, ZODIA, AMBOR.
20 a$="STAR2.BIN":ad=&100:





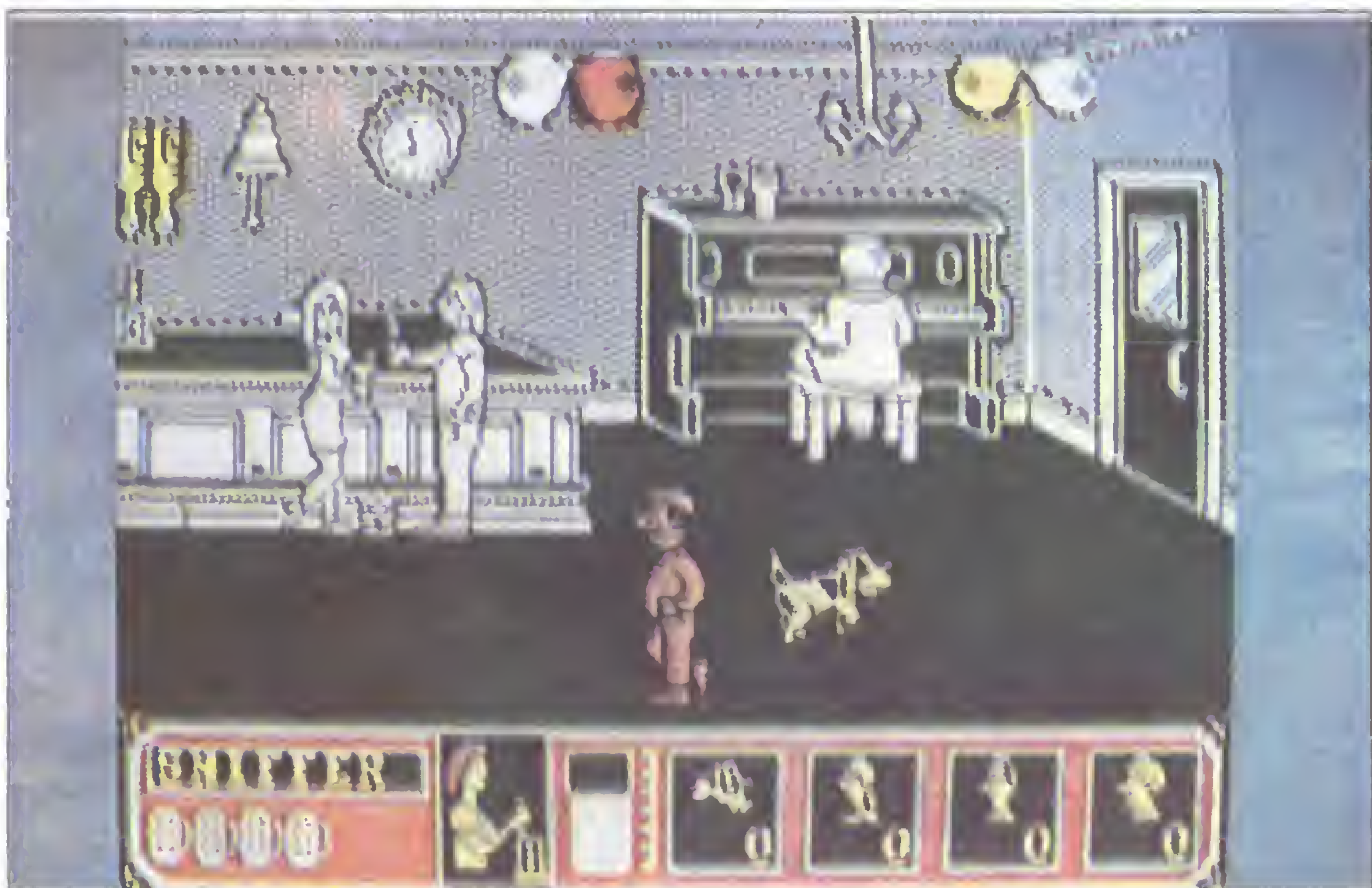


le=&A1A2:ex=0:sum=12801
130 DATA 175,50,246,28,50,69,46,50,140,64,201

Gry **RAMBO** i **COBRA-STALLONE** są tak trudne, że nawet dla wirtuozów klawiatury i joysticka pomyślne ich ukończenie jest praktycznie niemożliwe. Jednak po poprawieniu trzeciego pliku programu RAMBO (adres umieszczenia &1000, długość &9000) uzyskamy całkowitą odporność na kolizje z wrogiem. W trakcie wędrówek po dżungli znajdujemy broń, którą możemy użyć po wybraniu jej klawiszem SPACE.

20 a$="RAMBO1.BIN":ad=&1000:le=&9000:ex=0:sum=12559
130 DATA 6,34,33,89,158,54,0,35,16,251,201

Akcja gry **COBRA** firmy Ocean toczy się w mieście, w którym banda maniakalnych morderców chce zabić piękną Ingrid, a naszym zadaniem jest uratowanie dziewczyny. Na początku gry dysponujemy jedynie własnymi pięściami i musimy robić z nich użytek aż do czasu gdy zdobędziemy broń, której należy szukać w oknach domów. Po wyeliminowaniu snajpera lub po prostu zbiciu szyby może ukazać się w tym oknie kanapka, która zamieni się w jedną z dostępnych broni - karabin maszynowy, pistolet lub nóż. Wyboru broni dokonujemy klawiszami kursora (góra, dół).



Wprowadzenie poprawek do trzeciego pliku programu o nazwie COBRA1.BIN (adres umieszczenia &40, długość &A6B0) umożliwi nam pomyślne ukończenie gry.

20 a$="COBRA1.BIN":ad=&40:le=&A6B0:ex=0:sum=12388
130 DATA 175,50,202,78,201

Ciekawe rozwiązania graficzne i dobrą animację prezentuje **FIGHTING WARRIOR** firmy Melbourne House. Wszyscy, których pasjonuje walka na miecze, mogą ułatwić sobie zadanie uzyskując nieśmiertelność w tej grze poprzez zmianę trzeciego pliku programu o nazwie WARRIOR1.BIN (adres umieszczenia &3C6, długość &A2B6).

20 a$="WARRIOR1.BIN":ad=&3C6:le=&A2B6:ex=3C6:sum=12543
130 DATA 62,192,50,215,141,201

W oparciu o film Spielberga powstała gra **GOONIES**. Grupa urwisów (w programie jest ich dwóch) szuka skarbu. Ułatwiamy sobie zadanie modyfikując plik GOONIES.BIN (adres umieszczenia &3E8, długość &9E3E).

20 a$="GOONIES.BIN":ad=&400:le=&9E3E:ex=0:sum=12219
130 DATA 175,50,40,71,201

Również w oparciu o film powstała gra **BIGGLES**. Biggles to nazwisko pilota samolotu z okresu I wojny światowej, który zostaje uwikłany w perypetie z dziurą czasową i wraz ze swoim przyjacielem z XX wieku stara się rozwiązać zagadkę nowej niemieckiej broni.

Aby ukończyć grę, trzeba spełnić trzy warunki. Musimy uciec przed policją po londyńskich dachach; musimy dotrzeć do bunkra znajdującego się na poligonie, gdzie badana jest tajna broń; musimy także dolecieć samolotem do emitera infradźwięków - nowej niemieckiej tajnej broni. Po zmodyfikowaniu pliku programu BIGGLES.BIN (adres umieszczenia &100, długość &A573, adres wykonania &450) program nie będzie zauważał naszych wpadek.

20 a$="BIGGLES.BIN":ad=&100:le=&A573:ex=450:sum=12777
130 DATA 175,50,144,17,50,89,35,50,237,47,201

Ciekawą pozycją wśród programów zręcznościowych jest **BREAKTHRU**. Supersamochodem, dobrze uzbrojonym i potrafiącym wykonywać nawet długie skoki w powietrzu, wyruszamy do wielkich zakładów przemysłowych z misją wywiadowczą. Nasz samochód będzie skakał jak kangur po każdym naciśnięciu klawisza ">" (klawisz większość), a program nie będzie liczył naszych wpadek, jeżeli poddamy modyfikacji trzeci blok o nazwie BREAKT1.BIN (adres umieszczenia &1000, długość &8100).

20 a$="BREAKT1.BIN":ad=&40:le=&8100:ex=0:sum=12192
130 DATA 175,50,81,3,201

Na zakończenie tej długiej serii poprawek do gier dla komputerów Amstrad-Schneider, popularny i lubiany **ARKANOID** wprowadzony przez firmę Imagine-Taito. Jedyną wadą tej atrakcyjnej gry jest duży stopień trudności w niektórych sytuacjach, gdzie standardowy limit błędów praktycznie uniemożliwia przejście do dalszych etapów. Modyfikacja trzeciego bloku ARKANOID.NO2 (adres umieszczenia &40, długość &75C0) usunie tę niedogodność z gry.

20 a$="ARKANOID.N02":ad=&40:le=&75C0:ex=0:sum=12353
130 DATA 175,50,243,2,201

Commodore 64

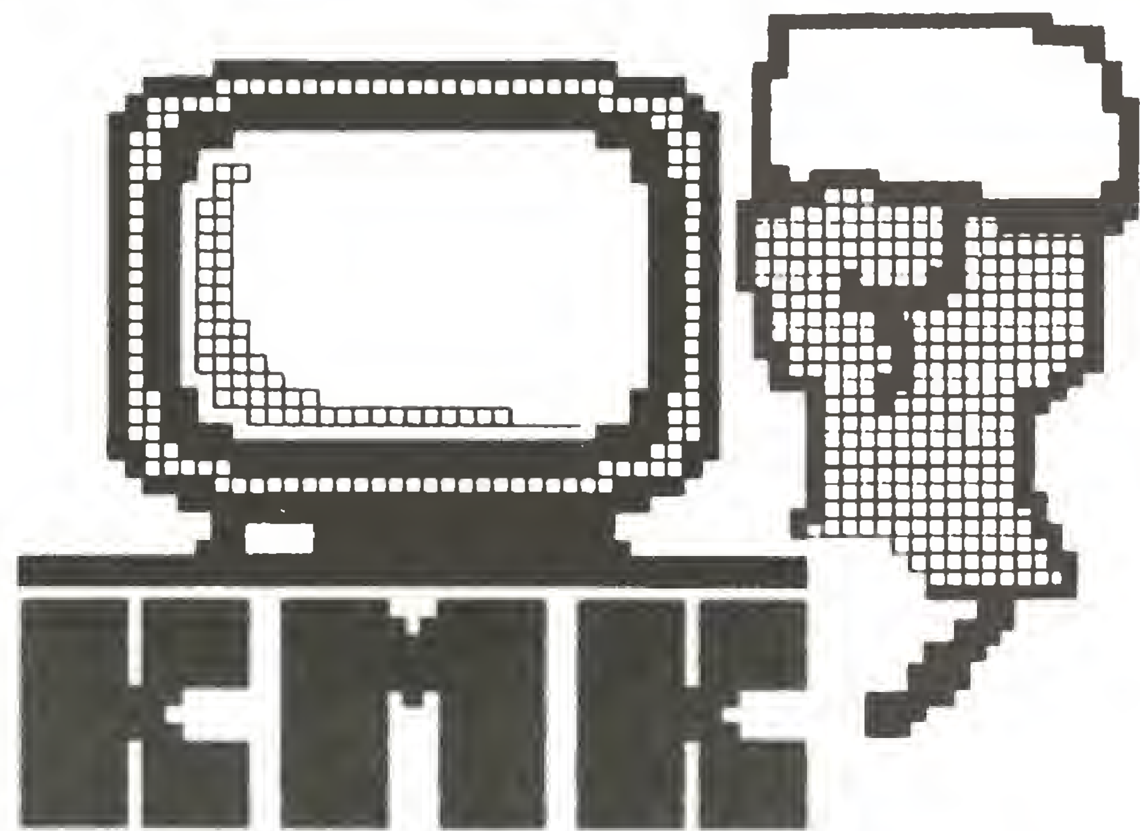
Marcin Topolewski przygotował poprawki do dwóch gier. Oto one.



BURNIN RUBBER firmy COLD-SOFTWARE zaliczamy do gier zręcznościowych. Jej fabuła jest bardzo prosta. Nasze zadanie, to zdobycie jak największej liczby punktów. Sterujemy samochodem, który porusza się po ulicy w tłumie innych pojazdów. Kierowanie odbywa się za pomocą joysticka (przód - przyśpieszanie; tył - hamowanie; prawo, lewo - kierunek). Oprócz tego mamy możliwość skakania (po naciśnięciu fire). Punkty zdobywamy pomagając innym pojazdom rozbić się na poboczu. W zależności od tego co rozbijemy, otrzymujemy od 100-500 punktów. Po ukończeniu każdej planszy otrzymujemy premię. Jej wysokość zależy od liczby rozbitych przez nas samochodów. Dla niecierpliwych podaję POKE na nieskończoną liczbę samochodów:
POKE 18432,173.

Każdy z nas już na pewno zna ze słyszenia lub z własnej praktyki program **DEFENDER** firmy Softfarm. Jest to jedna z wielu gier polegająca jedynie na strzelaniu. Przeciwnikami w tej grze są kosmiczne roboty. Naszym zadaniem jest obrona małych prostokątów, znajdujących się na dole ekranu. W swoim arsenale posiadamy laser i bomby jednorazowego użytku (niszczą wszystkie wrogie statki w zasięgu ekranu). Na górze ekranu znajduje się radar pomagający w nawigacji. Po każdych 10 000 punktów otrzymujemy nowy statek. Wystarczy jednak wpisać
POKE 3005,5,
by stać się niezależnym od dodatków.





PĘTLICZEK - bo pętla jest podstawą programowania. Tu znajdziesz kolejną porcję zadań naszego Klubu Mistrzów Komputera.
MĘTLICZEK - bo znajdziesz tu różne różności, związane z minikomputerem tak cienką nitką, że Redakcja już nie bierze za nią odpowiedzialności.
Redakcja strony klubowej: Leszek Rudak (ASCII 01), Adam Nowicki (ASCII 02).

ZADANIA KLUBOWE

28/88. Trójkąt liczbowy, pokazany niżej, powstaje przez wypisywanie kolejnych liczb naturalnych w coraz dłuższych kolumnach.

						37	
					26	38	
				17	27	39	
			10	18	28	40	
		5	11	19	29	41	
	2	6	12	20	30	42	
1	3	7	13	21	31	43	..
	4	8	14	22	32	44	
		9	15	23	33	45	
			16	24	34	46	
				25	35	47	
					36	48	
						49	

Tablica taka ma wiele zadziwiających własności (patrz "Delta" nr 12/82). Aby zobaczyć te własności, proponuję napisać program, który wypisze na ekranie dowolny fragment takiej tablicy (np. fragment wokół zadanej liczby).

29/88. Można wykazać, że dla każdej liczby n, ciąg postaci: n1 = suma kwadratów cyfr liczby n, n2 = suma kwadratów cyfr liczby n3, ... prowadzi zawsze do cyklu lub do jedynki, tzn. albo pewna liczba w tym ciągu będzie równa n, albo od pewnego miejsca zawsze otrzymywać będziemy 1.

Proponuję napisać program znajdujący ciąg o podanej konstrukcji dla dowolnej liczby n, wczytywanej jako dana dla programu.

30/88. Czasami musimy odejść od komputera na małą chwilkę. Nie warto wtedy wyłączać naszej maszyny. Niestety wyświetlanie ciągle tego samego obrazu powoduje szybsze zużycie monitora.

Proponuję napisać program wygaszający ekran monitora, gdy przez dłuższą chwilę nie został naciśnięty żaden klawisz.

WYLICZANKA

Weszliśmy w epokę komputerów. Na dowód niech posłuży fakt umieszczenia w zbiorze wyliczanek dziecięcych wyliczanki następującej:

"W komputerze siedzi zwierzę,
ma niebieskie kły,
wiesza pranie na ekranie,
pomagasz mu ty."

Autor zbioru "Ene due rabe" (Wydawnictwo Dolnośląskie, Wrocław 1987), z którego pochodzi ta wyliczanka zapewnia, że wszystkie wyliczanki pochodzą od dzieci ze szkół bolesławickich i wrocławskich. Niestety nie podano, do jakich zabaw stosowana jest wyliczanka o komputerze. Myślę, że powstała ona w jakiejś szkole, która dostała komputer. Dzieci odliczają i na kogo wypadnie, ten może użyć szkolnego komputera.

FRAKTAL

Co przedstawia ten rysunek?

Jest to FRAKTAL.

Odpowiedź jest prawdziwa, choć bezużyteczna. Któż bowiem wie czy fraktal to zwierzę, czy drzewo? Oczywiście są ludzie, którzy wiedzą, co to jest i do czego służy FRAKTAL. Ci ludzie są zwykle matematykami albo informatykami i na co dzień usiłują nauczyć komputer rozpoznawania obrazów. Fraktale dają nadzieję na spełnienie ich marzeń.

Trudno w paru słowach wyjaśnić, co to jest fraktal. W dużym uproszczeniu można powiedzieć, że fraktal jest figurą, która powstaje z siebie samej poprzez pomniejszenie i zwielokrotnienie.

Popatrzmy na rysunek. Trójkąt składa się z trzech mniejszych, odpowiednio ułożonych trójkątów. Mniejsze trójkąty też są zbudowane z mniejszych, a te z jeszcze mniejszych itd.

Najciekawsze, a zarazem najważniejsze jest to, że program rysujący fraktala jest bardzo prosty! Nie będę dziś wyjaśniał dlaczego. Zachęcam do eksperymentów na gotowym programie (program zamieszczony poniżej rysuje przedstawionego fraktala). O fraktalach i programach generujących je pomówimy wkrótce (kolega obiecał mi program rysujący dowolne fraktale, a istnieją bardzo ciekawe...).

Przedstawiony program przeznaczony jest dla IBM PC z CGA, ale łatwo przerobić procedury graficzne na dowolną implementację Pascala.

```
program fraktal;

const {wartości zalezą od implementacji}
        x1=0;       y1=199;
        x2=639; y2=199;
        x3=320; y3=0;

var x,y,i:integer;

procedure poczatek;
    {inicjacja trybu graficznego}
begin
    {treść zalezy od implementacji}
   HiRes
end;

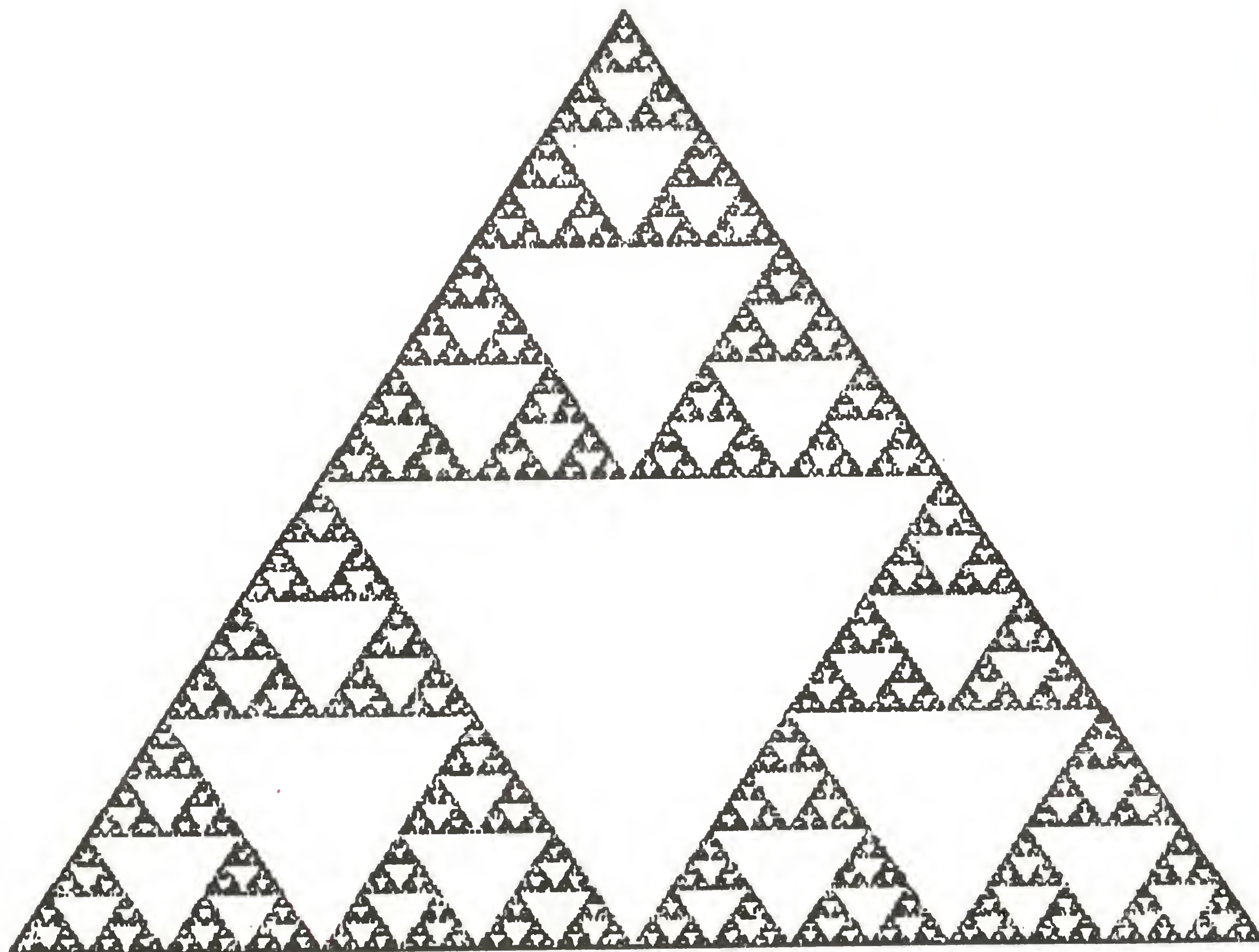
procedure punkt(a,b:integer);
    {rysowanie punktu o współrzędnych a i b}
begin
    {treść zależy od implementacji}
   Plot(a,b,1)
end;

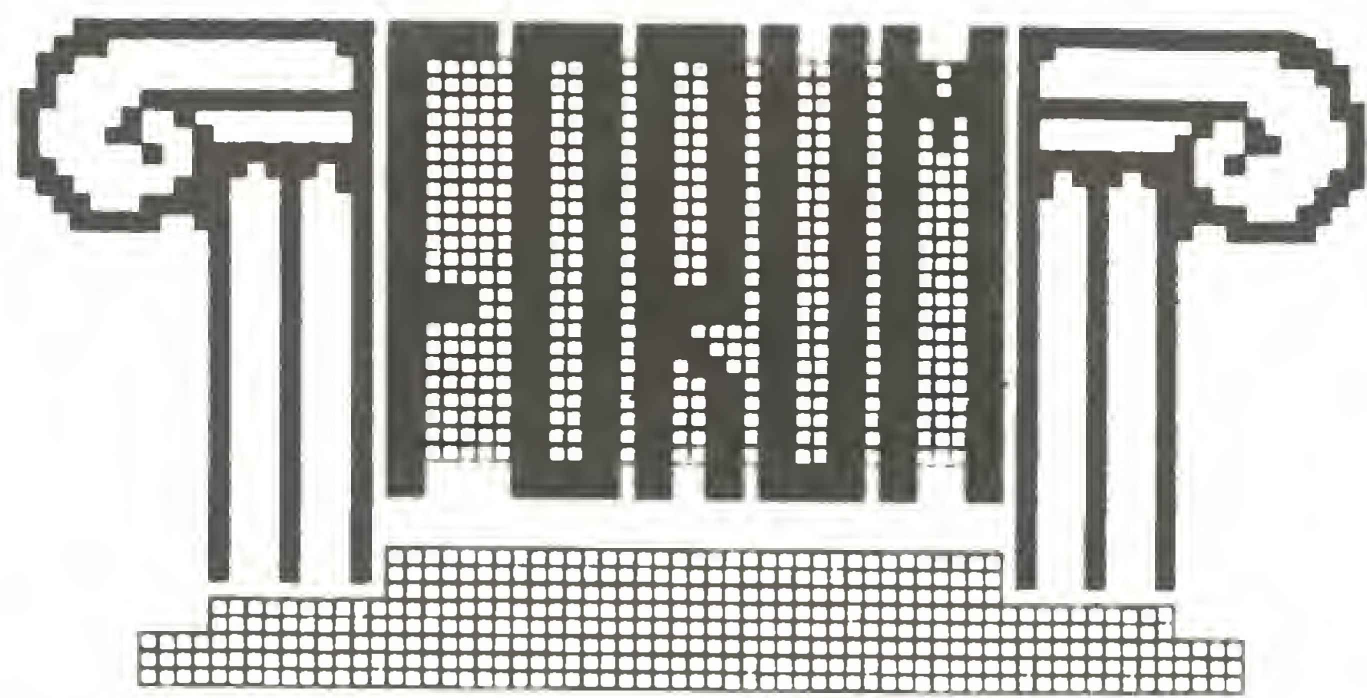
procedure linia(a1,b1,a2,b2:integer);
      {rysowanie linii od punktu
        (a1,b1) do punktu (a2,b2)}
begin
    {treść zależy od implementacji}
   Draw(a1,b1,a2,b2,1)
end;

procedure koniec;
    {zamknięcie trybu graficznego}
begin
    {treść zalezy od implementacji}
   TextMode(BW80)
end;

function los(a:integer):integer;
    {funkcja dająca całkowitą liczbę
      losową z przedziału od 1 do a}
begin
    {treść zalezy od implementacji}
   los:=Random(a)
end;

begin
   x:=300;
   y:=100;
   poczatek;
   linia(x1,y1,x2,y2);
   linia(x2,y2,x3,y3);
   linia(x3,y3,x1,y1);
   for i:=0 to 30000 do begin
     case los(3) of
      0: begin
             x:=x+((x1-x) div 2);
             y:=y+((y1-y) div 2);
          end;
      1: begin
             x:=x+((x2-x) div 2);
             y:=y+((y2-y) div 2);
          end;
      2: begin
             x:=x+((x3-x) div 2);
             y:=y+((y3-y) div 2);
          end;
     end;
     punkt(x,y);
   end;
   repeat until keypressed;
   koniec;
end.
```





Drodzy Czytelnicy!

"Forum" to rubryka przeznaczona w całości do Waszej dyspozycji. Możecie do niej pisać nie tylko o swoich osiągnięciach, nadsyłać użyteczne programiki czy "sztuczki i chwyty", które odkryliście, ale możecie informować także o problemach, które spotykacie w pracy z mikrokomputerami. Być może ktoś inny je rozwiązał i będzie mógł Wam tą drogą pomóc.

Jeżeli przedmiotem korespondencji jest program, to prosimy czytelników o załączanie (w miarę możliwości) dwóch jak najbardziej kontrastowo (tzn. czarno na białym) wydrukowanych listingów programu. Gdy program jest napisany w asemblerze i autor podaje listę odpowiednich POKE'ów, dobrze jest zaopatrzyć je w sumę kontrolną, która ułatwi potem innym uruchomienie programu.

Poza tym prosimy o zwięzłe formułowanie listów, o umieszczanie na kopertach dopisku "Forum" oraz podawanie wewnątrz listu dokładnego adresu nadawcy. Za publikowane w tej rubryce programy i ciekawe "sztuczki i chwyty" przysługuje honorarium, zgodne z obowiązującymi u nas stawkami. Dla przypomnienia podajemy nasz adres:

PMI "Komputer"
ul. Koszykowa 6a
00-564 Warszawa
"Forum"

Dzisiaj prezentujemy: jak ułatwić używanie popularnego edytora "ChiWriter" (komputery zgodne z IBM PC), drobne usprawnienie kompilatora języka Pascal (ZX Spectrum), program w Turbo Pascalu umożliwiający ładowanie ekranu nagranego na dysku, parę użytecznych programików dla Commodore 64, chwyt ułatwiający odbezpieczenie programu w Simons'Basic (Commodore 64), odpowiedź na pytanie z numeru 5/88 "Jak odczytać z poziomu asemblera położenie joysticka", programik umożliwiający dzielenie dwóch liczb z dowolną dokładnością (C116) oraz ulepszenie programu Reset dla Atari XL/XE z nr. 5/88.

Wszystkich zainteresowanych serdecznie zapraszamy do współpracy.

Redakcja

Użyteczne programiki (Commodore 64)

Szanowna Redakcjo!

Mam 15 lat i jestem Waszym stałym czytelnikiem. Informatyką zajmuję się czwarty rok. Korzystam z komputera Commodore C64. Przesyłam moje własne cztery programy w języku wewnętrznym, którymi chciałbym się podzielić z Redakcją i Czytelnikami.

Pierwszy program powoduje, że kursor na ekranie C64 przestaje migać. Do napisania tego programiku skłoniło mnie to, że dłuższe wpatrywanie się w ekran z "migającym" kursorem zaczyna być męczące dla oczu.

Program drugi zmienia dotychczasową rolę klawiszy RUNSTOP i RESTORE. Po uruchomieniu tej procedury, jednoczesne naciśnięcie tych klawiszy spowoduje tzw. reset pamięci. Zaletą tego programiku jest to, że funkcja RESET jest nadal aktywna po jej użyciu. Dzięki tej procedurze nie trzeba dorabiać (sprzętowo) przycisku RESET.

Program trzeci sprawia, że klawisze funkcyjne F1 i F3 są odpowiedzialne za kolor ramki (F1) i tła (F3). Aktualny kolor tła czy ramki zależy od położenia kursora. To znaczy: jeżeli kursor jest w wierszu 1 i 2 kolumnie, to wciśnięcie F1 równoznaczne będzie z operacją: POKE 53280,1, a F3 - POKE 53281,2.

Program czwarty jest bardzo przydatny dla tych, którzy wpisując program nie patrzą na ekran, lecz na klawiaturę. Za pomocą sygnału dźwiękowego program ten ostrzega o zbliżaniu się kursora do prawej krawędzi ekranu.

Wszystkie te programy napisane zostały na komputerze Commodore 64. Programy wykorzystują wewnętrzny system przerwań komputera. Poza drugim, wszystkie zmieniają do własnych celów wektor o adresach 788 i 789. Programy te, za wyjątkiem RESET, nie są relokowalne i nie mogą działać jednocześnie.

Łączę wyrazy szacunku
Radosław P.Grabiec
Warszawa

```
0 REM      << NOWY KURSOR ($02A7) >>
1 REM      << RADOSLAW.P.GRABIEC  >>
2 REM      <<     1988 - GCS.     >>
3 :
5 SK=0:TEM=679
10 FORI=TEM TO TEM+48:READ D:POKEI,D
20 SK=SK+D:NEXT
25 IF SK<>6785 THEN PRINT"ERROR":END
30 SYSTEM
35 :
40 DATA 120,169,180,160,002,141,020,003
45 DATA 140,021,003,088,096,032,234,255
50 DATA 165,204,208,026,164,211,165,207
55 DATA 208,020,230,207,032,036,234,177
60 DATA 209,133,206,073,128,145,209,177
65 DATA 243,173,134,002,145,243,076,097
70 DATA 234
```

1

```
100 REM     <<  RESET - ($033C)    >>
110 REM     << RADOSLAW P.GRABIEC >>
111 REM     <<     1988 - G.C.S.   >>
115 :
120 SK=0:A=828:S%=A/256:M%=A-256*S%
130 FOR I=A TO A+21:READ Q:POKEI,Q
140 SK=SK+Q:NEXT
150 IF SK<>2845 THEN PRINT"ERROR":END
160 POKE792,M%:POKE793,S%
165 :
200 DATA 169,254,160,246,162,014,141
210 DATA 032,208,140,033,208,142,134
220 DATA 002,032,068,229,032,148,227
230 DATA 064
```

2

```
1 REM     <<  KOLORY - ($02A7)    >>
3 REM     << RADOSLAW P.GRABIEC  >>
5 REM     <<     1988 - GCS.      >>
7 :
10 SK=0:TEM=679
15 FORI=TEM TO TEM+41:READ Q:POKEI,Q
20 SK=SK+Q:NEXT
25 SYSTEM
30 IF SK<>5013 THEN PRINT"ERROR":END
35 :
40 DATA 120,169,180,160,002,141,020,003
45 DATA 140,021,003,088,096,165,197,201
50 DATA 004,240,007,201,005,240,011,076
55 DATA 049,234,166,211,142,032,208,076
60 DATA 049,234,166,214,142,033,208,076
65 DATA 049,234
```

3

```
1 REM    << MARGINES - ($02A7) >>
2 REM    << RADOSLAW P.GRABIEC >>
3 REM    <<     1988 - GCS.     >>
4 :
5 SK=0: TEM=679
10 FORI=TEM TO TEM+43:READQ:POKEI,Q
15 SK=SK+Q:NEXT
20 IF SK<>4862 THEN PRINT"ERROR":END
25 SYSTEM
30 :
40 DATA 120,169,180,160,002,141,020,003
45 DATA 140,021,003,088,096,165,211,201
50 DATA 035,240,007,201,075,240,003,076
55 DATA 049,234,169,015,141,024,212,162
60 DATA 005,142,005,212,160,000,140,024
65 DATA 212,076,049,234
```

4

Usprawnienie ChiWritera (IBM PC)

Szanowna Redakcjo!

Popularny edytor tekstu ChiWriter ma kilka wad. Podstawową z nich jest powolność, częściowo wynikająca ze specyficznej organizacji edytora. Uruchamianie rozpoczyna się długim wczytywaniem do pamięci 20 zbiorów znaków (dla wersji 2.04), po czym należy wybrać rodzaj pracy z wyświetlonego na ekranie katalogu. Istnieją jednak możliwości wprowadzenia pewnych usprawnień tego procesu.

Niewielu użytkowników ChiWritera wie, iż wzorem innych edytorów pozwala on przy wywołaniu podać nazwę żądanego pliku jako parametr. Przykładowo, wywołanie do obróbki tekstu o nazwie ARTYKUL.CHI może wyglądać tak: cw artykul. Po wczytaniu zbiorów znaków edytor przystępuje od razu do czytania pliku z tekstem.

Dodatkowe usprawnienie pracy można osiągnąć wykorzystując możliwości ustalania tzw. konfiguracji edytora. Pojęcie konfiguracji obejmuje tu takie parametry jak: marginesy, odstępy między liniami, listę wykorzystywanych zbiorów znaków, jakość druku, sposób numeracji stron itp. Specjalna opcja w menu edytora (ENVIRONMENT/CONFIGURATION) pozwala wyświetlić na ekranie wszystkie te parametry (LIST), zapisać je do pliku dyskowego o rozszerzeniu .PAR (WRITE) lub wprowadzić z dysku do pamięci przez wczytanie odpowiedniego pliku z tym rozszerzeniem (READ).

ChiWriter rozpoczyna pracę od przeczytania pliku CONFIG.PAR, zawierającego informacje o domyślnych (default) wartościach parametrów. Jeśli plik ten będzie zawierał parametry odpowiednie dla naszego tekstu, w tym listę zbiorów znaków ograniczoną do niezbędnych, wówczas rozpoczynając pracę edytor wczyta tylko te właśnie zbiory, a pozostałe parametry automatycznie ustawi na odpowiednie wartości. Pozwala to znacznie skrócić oczekiwanie na gotowość do pracy oraz oszczędzić mozolnego ustawiania wszystkich parametrów.

Problem powstaje w momencie, gdy z edytora korzysta wielu użytkowników lub jeden, pracujący nad różnymi tekstami. Przy każdej zmianie przetwarzanego tekstu parametry konfiguracji również będą wymagały zmiany za pomocą opcji menu ENVIRONMENT/CONFIGURATION/READ. Można jednak tego uniknąć wywołując każdorazowo edytor przez odpowiedni program wsadowy (typu "batch"). Jedynym warunkiem jest, aby plik zawierający parametry konfiguracji dla danego tekstu posiadał identyczną z nim nazwę, różniąc się tylko rozszerzeniem dodanym przez edytor (tekst - rozszerzenie.CHI, parametry - rozszerzenie .PAR). Program o nazwie np. CW.BAT, co zapewnia identyczność wywołania, wygląda następująco:

```
echo off
c:
cd\chiwrite
if not exist %1.par then goto NIEMA
copy %1.par config.par
goto CHI
:NIEMA
copy standard.par config.par
:CHI
cw %1
c\d
```

Program przyjmuje założenie, że wszystkie pliki edytora znajdują się na dysku c:, w katalogu \ CHIWRITE. Wykorzystuje się tu możliwość przyjmowania przez programy typu "batch" parametrów, które - zależnie od ich liczby - oznacza się kolejno: %1, %2, %3 itd. W naszym przypadku jedyny parametr, jakim jest nazwa przetwarzanego pliku, występuje jako %1.

Działanie programu najlepiej prześledzić na przykładzie. Załóżmy, że chcemy przetwarzać tekst zawarty w pliku ARTYKUL.CHI. Po wywołaniu - cw artykul - łańcuch znaków "artykul" podstawiany jest w programie CW.BAT na miejsce symbolu %1. Następuje sprawdzenie, czy w katalogu bieżącym istnieje plik ARTYKUL.PAR (w programie symbol %1.par). Jeżeli tak, to jest on kopiowany ze zmianą nazwy na CONFIG.PAR, po czym następuje wywołanie edytora z parametrem "artykul" (%1). W ten sposób, w chwili gdy Chiwriter zaczyna czytać informacje o konfiguracji z pliku CONFIG.PAR, zawiera on już dane dotyczące wybranego tekstu. Po wprowadzeniu wszystkich potrzebnych danych edytor doda do nazwy tekstu domyślne rozszerzenie .CHI i automatycznie przystąpi do wczytywania z dysku odpowiedniego pliku.

W przypadku gdy zamierzamy dopiero zacząć pisanie tekstu i plik konfiguracyjny ARTYKUL.PAR jeszcze nie istnieje, pod nazwę CONFIG.PAR kopiowana jest zawartość pliku STANDARD.PAR, w którym umieszczono zestaw parametrów przyjętych jako standardowy, m.in. uwzględniający wszystkie 20 zbiorów znaków pisarskich. Rozwiązanie to zapewnia ponadto, że w przypadku wywołania edytora bez podania nazwy tekstu plik konfiguracyjny będzie zawierał najbardziej uniwersalny zestaw parametrów. Jako uzupełnienie dodać należy, że eliminacji zbędnych zbiorów znaków (fonts) dokonać można usuwając je z pamięci operacyjnej za pomocą opcji menu ENVIRONMENT/FONT/DELETE przed zapisaniem do pliku danych o konfiguracji.

Z poważaniem
Stanisław Zaremba
Gliwice

Odbezpieczenie programu w Simons'Basic (Commodore 64)

Szanowna Redakcjo!

Przesyłam krótki program użytkowy, który może być przydatny użytkownikom programu Simons'Basic dla komputerów Commodore 64. Simons'Basic umożliwia zabezpieczenie programu napisanego w Basicu poprzez wpisanie instrukcji DISAPA na początku zabezpieczonego wiersza oraz wykonanie komendy SECURE 0 w trybie bezpośrednim. Od tej pory program przy listowaniu pokazuje jedynie numery wierszy. Oczywiście należy przechowywać kopię programu, gdyż po zabezpieczeniu nie można normalną drogą (tj. poprzez podanie innej komendy) odbezpieczyć go. Jeżeli ktoś pragnie jednak program odbezpieczyć, można to uczynić poprzez wpisanie poniższego programiku przed program główny:

```
1 S=2049
2 IF PEEK(S+4)=0 THEN POKE S+4,100: POKE S+5,65
3 S=PEEK(S)+PEEK(S+1)*256
4 IF PEEK(S+1)<>0 THEN 2
5 END
```

Po uruchomieniu program przywraca sytuację, jaka miała miejsce po wpisaniu instrukcji DISAPA, ale przed użyciem SECURE 0. Teraz można dokonywać poprawek (np. usunąć wszystkie DISAPA i program będzie odbezpieczony). Działanie programiku opiera się na wyszukaniu wszystkich miejsc, w których jest zakodowana instrukcja DISAPA i podstawieniu kodu instrukcji "widzialnej".

Swoją drogą, jak marne jest to zabezpieczenie, skoro tak prosto można je zlikwidować.

Piotr Górniak
Wrocław

Dokładne dzielenie (Commodore 116)

Droga Redakcjo!

Jestem właścicielem komputera Commodore 116. Pracuję już na nim przeszło dwa lata i jak na razie nie narzekam. Sporo programów układam sam, zwłaszcza edukacyjnych. Chciałbym się podzielić z czytelnikami dość krótkim, ale efektownym programem dotyczącym dzielenia. Operację dzielenia wykonuje on z zadaną dokładnością.

Jedynym ograniczeniem w liczbie miejsc po przecinku jest sam ekran. Natomiast jeżeli ktoś dysponuje drukarką, program może liczyć nawet do kilku tysięcy miejsc po przecinku.

Robert Orczykowski
Inowrocław

```
1 REM *** DOKLADNE DZIELENIE ***
2 REM *** R.ORCZYKOWSKI ***
3 R=6: Z=1
4 PRINT"DO ILU MIEJSC PO PRZECINKU
  MAM LICZYC":INPUT K
5 PRINT"PODAJ DZIELNA":INPUT X
6 PRINT"PODAJ DZIELNIK":INPUT Y
7 SCNCLR: PRINT X"/"Y" = "
8 IF X<Y THEN A=0: GOTO 10
9 A=INT(X/Y):IF A=1 THEN END
10 PRINT A",";
11 X=(X-A*Y)*10: A=INT(X/Y)
12 L=L+1:A$=STR$(A):R=R+1
13 IF R/41=INT(R/41) THEN Z=Z+1:R=1
14 CHAR,R-1,Z,RIGHT$(A$,1)
15 IF L=K THEN END: ELSE 70
```

Reset (Atari XL/XE) ("Forum", 5/88, str.28)

Szanowna Redakcjo!

Program, który chcę przedstawić, jest ulepszoną wersją programu Andrzeja Lecha z rubryki "Forum" w numerze 5/88.

Różnica polega na innej realizacji programu w kodzie maszynowym. Dzięki wprowadzonej zmianie długość programu zmalała do jednej trzeciej.

Jest to przykład jak dużo można zyskać stosując odpowiednią strukturę programu.

Maurycy Mikulski
Gdynia

```
10 FOR I=1 TO 38: READ A: S=S+A: NEXT I
20 IF S<>2718 THEN ?"BLAD DANYCH": END
30 RESTORE
40 FOR I=1536 TO 1573: READ A: POKE I,A: NEXT I
50 POKE 2,0: POKE 3,6: POKE 9,2: NEW
60 DATA 162,21,189,16,6,157,186,156,202,208,247,
   169,13,141,74,3,96
70 DATA 48,47,43,37,0,24,20,18,12,17,18,26,31,2,
   125,2,27,26,50,53,46
```

Ładowanie ekranu w Turbo Pascalu (CPC6128)

Przedstawiona poniżej procedura umożliwia wyświetlenie na ekranie monitora obrazu nagranego uprzednio na dyskietce. Może on być utworzony i zapisany na dysku z poziomu dowolnego języka programowania lub edytora graficznego - np. Art Studio (opcje Palette i Compress wyłączone), Screen Designer.

Przed jej wywołaniem należy ustawić właściwy tryb pracy monitora (MODE) oraz zdefiniować paletę kolorów (INK) wykorzystywanych w ładowanym obrazie. Para-

> 24

```
type string12 = string[12];

procedure LoadScreen(zbior : string12);

{------------------------------------------------------------}
{ Procedura odczytuje obraz ze zbioru na dyskietce           }
{ i zapisuje go do pamieci obrazu.                           }
{ Parametr wejsciowy: nazwa zbioru w formacie "aaaaaaaa.aaa" }
{ Zbior musi znajdowac sie w biezacym napedzie dyskowym      }
{------------------------------------------------------------}

const
     codedl = 30;
     code : array [1..codedl] of byte=
               ($f3,$3e,$c1,$0e,$2,$6,$7f,$ed,$79,
                $21,0,0,$11,$00,$00,1,$0,$02,$ed,$b0,
                $3e,$c2,$0e,$2,$6,$7f,$ed,$79.$fb,$c9);
           { kod podprogramu transmisji danych z bufora do pamieci obrazu }
var
     c1    : array [1..codedl] of byte;
           { obszar, w ktorym bedzie znajdowal sie podprogram transmisji  }
     bufor : array [1..8192] of byte;
           { bufor na wczytywane z dysku dane - moze byc mniejszy !       }
     ekran : file;
     poczb, ilezb, gdzie : ^integer;
     i                   : integer;

begin
   inline($cd/$5a/$fc/$DB/$BB);            { wywolanie procedury CLG          }
   for i:=1 to codedl do c1[i]:=code[i];  { ladowanie podprogramu transmisji }
   poczb:=Ptr)Addr(c1[11]));               { inicjowanie adresow parametrow   }
   ilezb:=Ptr(Addr(c1[17]));               { transmisji                       }
   gdzie:=Ptr(Addr(c1[14]));
   poczb^:=Addr(bufor);                    { ustalenie parametrow transmisji  }
   ilezb^:=$2000;
   gdzie^:=$4000;
   assign(ekran,zbior);                    { przydzielenie zbioru             }
   reset(ekran);                           { otwarcie zbioru do odczytu       }
   BlockRead(ekran,bufor,1,i);             { pominiecie pierwszych 128 bajtow }
   BlockReda(ekran,bufor,64,i);            { wczytanie do bufora 8 KB         }
   inline($cd/c1);                         { przeslanie danych z bufora       }
   gdzie^ := gdzie^ + ilezb^;              {       do pamieci obrazu          }
   BlockRead(ekran,bufor,64,i);            { wczytanie do bufora 8 KB         }
   inline($cd/c1);                         { przeslanie danych                }
   close(ekran);                           { zamkniecie zbioru                }
end; { LoadScreen }
```

23 <

metrem wejściowym procedury jest dwunastoznakowa nazwa zbioru np.:"obraz.scr". Aby uniknąć błędów, jej definicję należy umieszczać w miarę możliwości na początku kodu źródłowego programu.

Przedstawioną procedurę można rozbudować o:

- możliwość podawania jako parametru pełnej nazwy zbioru z oznaczeniem napędu dyskowego,
- sprawdzanie czy zbiór z obrazem istnieje na dyskietce i w razie jego braku sygnalizowanie tego odpowiednim komunikatem,
- szybkie (niestopniowe) wyświetlenie obrazu po jego wczytaniu poprzez odpowiednią manipulację paletą kolorów.

Nie jest to jedyne i na pewno nie najlepsze rozwiązanie problemu, bowiem nie mam pełnej dokumentacji oprogramowania podstawowego zawartego w pamięci ROM.

W mojej pracy pomogły mi procedury opracowane przez kolegę MB.

Marian Zaborowski
Koszalin



Hisoft Pascal (ZX Spectrum)

Szanowna Redakcjo!

Jestem użytkownikiem mikrokomputera ZX Spectrum. Od niedawna próbuję programować w Pascalu. Hisoft Pascal jest wspaniałym narzędziem, a nieocenione usługi daje programik "PRINT 64". Jednak przy jego stosowaniu pojawiają się pewne niedogodności. Podane poniżej POKE-i na pewno oszczędzą kilku godzin pracy zwolennikom Pascala.

1. Likwidacja przerwy w wydruku, pojawiającej się w połowie wiersza przy PR64:

HP4S	HP4TM16
26048,0	26613,0
26049,0	26614,0
26050,0	26615,0
26064,0	26629,0
26065,0	26630,0
26066,0	26631,0

2. Rezygnacja z migającego kursora, związanego ze strukturą kratkową ekranu:

HP4S	HP4TM16
24837,0	24836,0

Powyższe poprawki znacznie ułatwiają pracę. Działają również bez PRINT 64. Należy je oczywiście wprowadzić po nagraniu programu głównego.

Z poważaniem
Wojciech Bartkowiak
Koszalin

Joystick a kod wewnętrzny Commodore 16 ("Forum" 5/88, str. 29)

Szanowna Redakcjo!

Chciałbym odpowiedzieć na pytanie postawione w liście Grzegorza Mańturzyka w nr. 5/88.

Położenie drążka sterowego z poziomu języka maszynowego najprościej można odczytać za pomocą procedury KERNAL, wywoływanej poleceniem JSR$FF9F. Po tej operacji w akumulatorze znajduje się wartość określająca, który klawisz naciśnięto. Na pewno każdy użytkownik C16 zauważył, że kiedy podłączymy drążek sterowy i zaczniemy nim manipulować, na ekranie pojawią się "dziwne" znaki. Manetka może więc działać jak klawiatura.

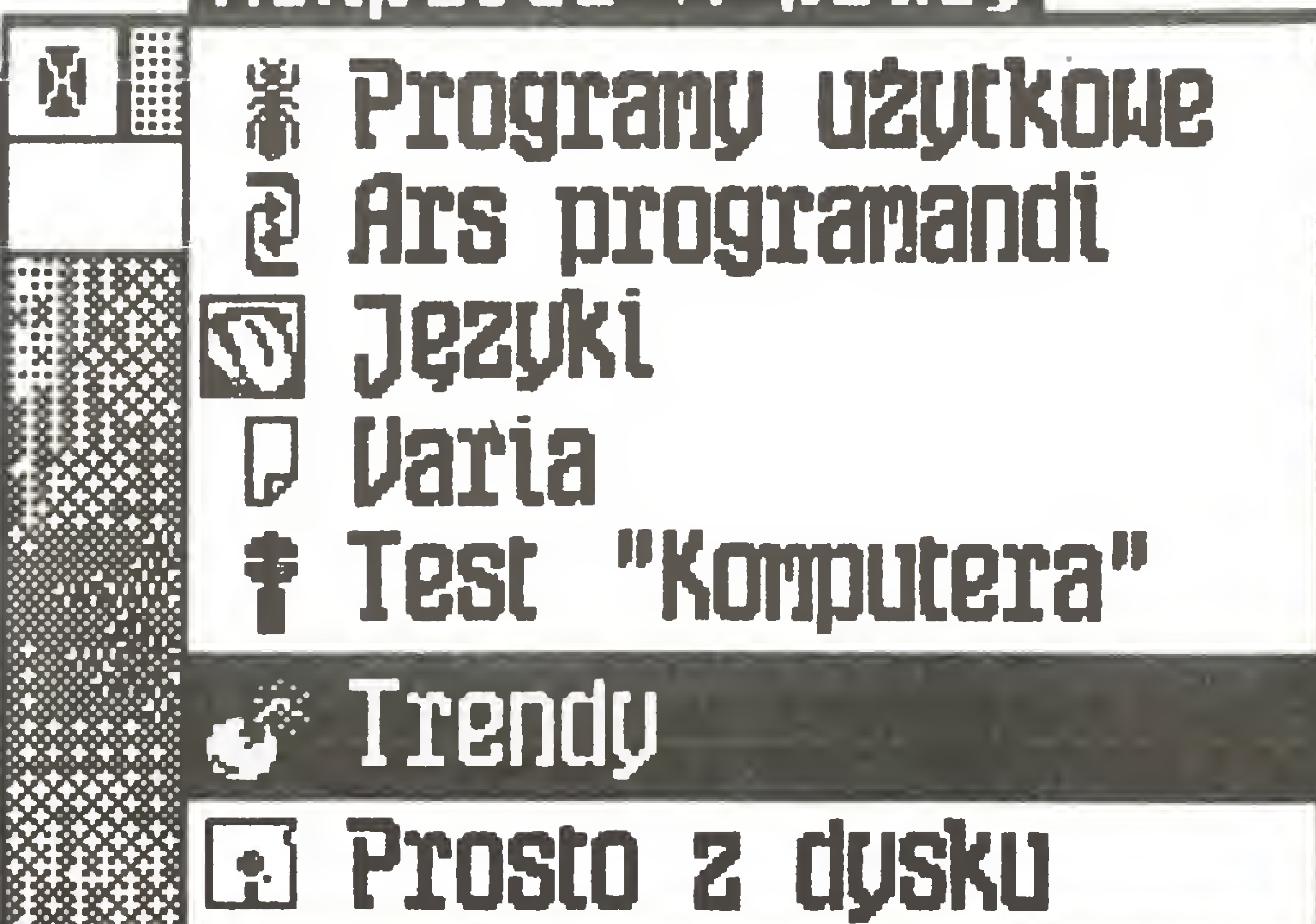
Przykładowy program może wyglądać następująco:

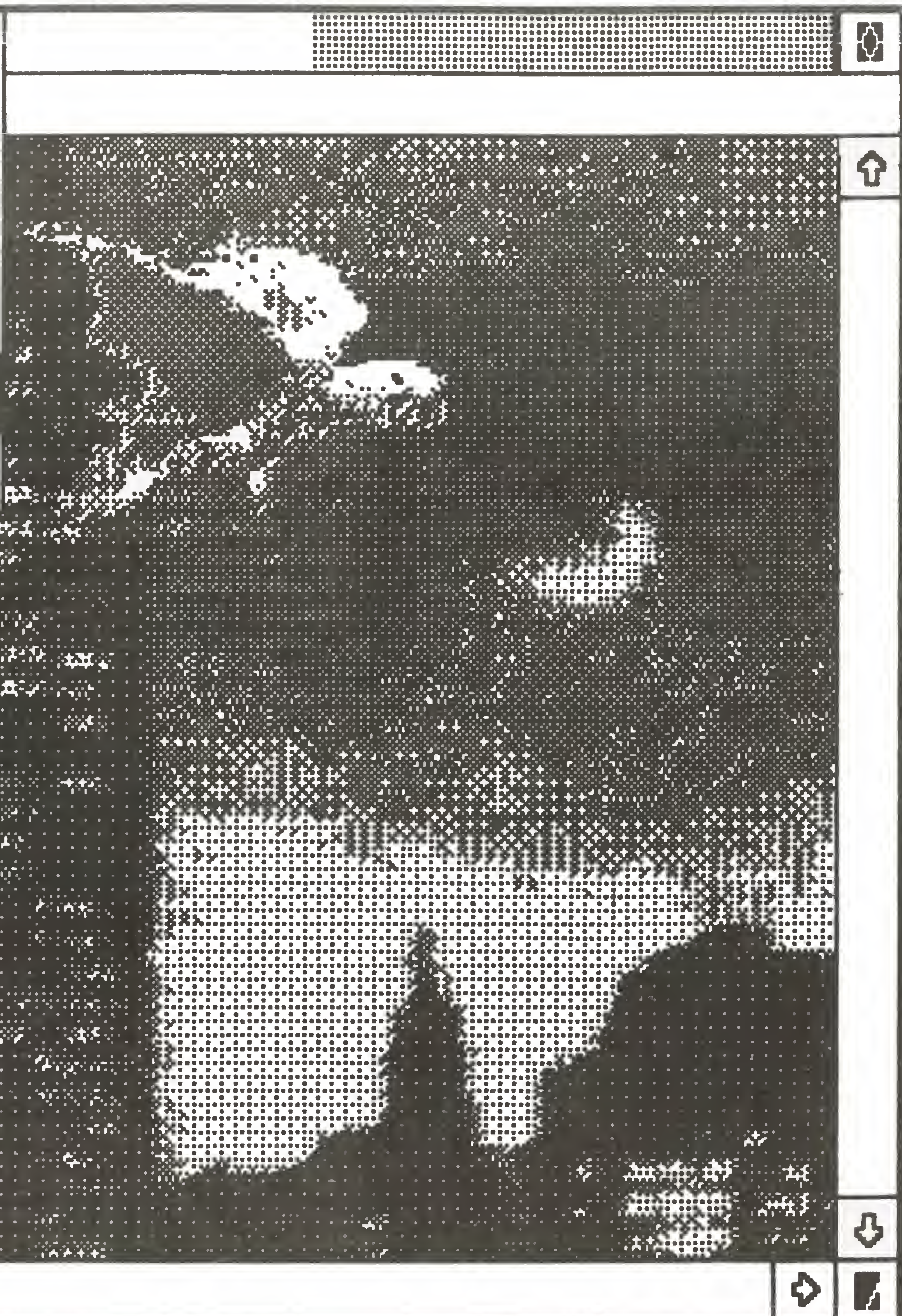
```
0609 JSR$FF9F
060C CMP$#51
060E BEQ$0613
0610 JMP$0609
0613 ...
```

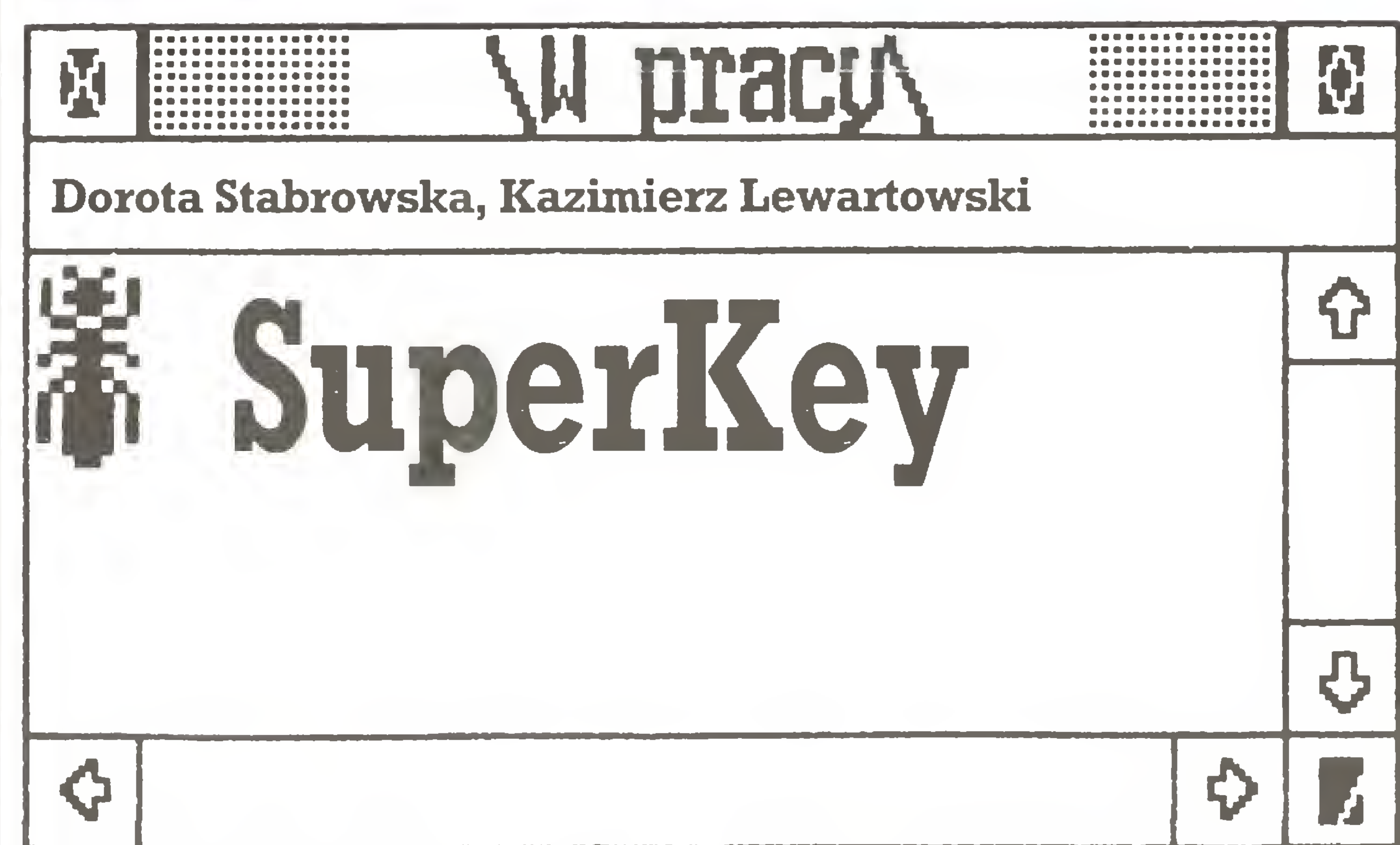
Programik ten sprawdza, czy naciśnięto "FIRE". Dopiero po spełnieniu tego warunku będą wykonywane instrukcje od komórki 0613.

Z kolei i ja mam pytanie: w jaki sposób napisać program obsługi pióra świetlnego?

Z poważaniem
Andrzej Szwabe
Bydgoszcz







Dorota Stabrowska, Kazimierz Lewartowski

SuperKey

Program: *SuperKey*
Producent: *Borland International Inc.*
Wersja: *1.15A*
Sprzęt: *IBM PC/XT/AT/Jr*
Pamięć: *min. 128K*
System: *PC-DOS 2.0 lub wyżej*

SuperKey należy do szeroko rozpowszechnionych w naszym kraju programów, jednak - zapewne z braku dokumentacji - dość rzadko używanych. Tymczasem zalicza się on do tej kategorii wyrobów Borlanda, które znacznie ułatwiają pracę z komputerem. Do głównych zadań programu należy: tworzenie i uruchamianie makrorozkazów do pracy z innymi programami, szyfrowanie dostępu do zbiorów, przeredagowywanie klawiatury oraz inne funkcje, o których poniżej. SuperKey jest programem rezydentnym, powinien być ładowany po wszystkich programach czuwających nie-Borlanda, **a przed** SideKickiem. Jeżeli używamy DOS 2.11, należy w **CONFIG.SYS** umieścić instrukcję **DEVICE=ANSI.SYS.**

Główne pliki dostarczane przez producenta to:

KEY	COM	41792	– główny plik programu;
KEYDES	COM	44912	– szyfrowanie dostępu do zbiorów;
KEYINST	COM	57354	– instalacja programu;
KEY	HLP	38400	– tekst ściągawki;
LAYOUT	COM	27136	– definiowanie klawiatury;
README	COM	16110	– program czytający plik READ-ME.KEY;
READ-ME	KEY	12245	– tekst uzupełniający podręcznik;
,	.MAC		– pliki zawierające przykładowe makrorozkazy;
,	.LAY		– pliki zawierające przykładowe definicje klawiatury.

SuperKey **musi** być uruchamiany z katalogu, w którym umieszczone są pliki **.COM** i **.HLP**. Po wypisaniu **C:KEY>KEY<Enter>** na ekranie ukazuje się etykieta programu, informacja o zainstalowanej pamięci, pamięci dostępnej po uruchomieniu SuperKey

> 26

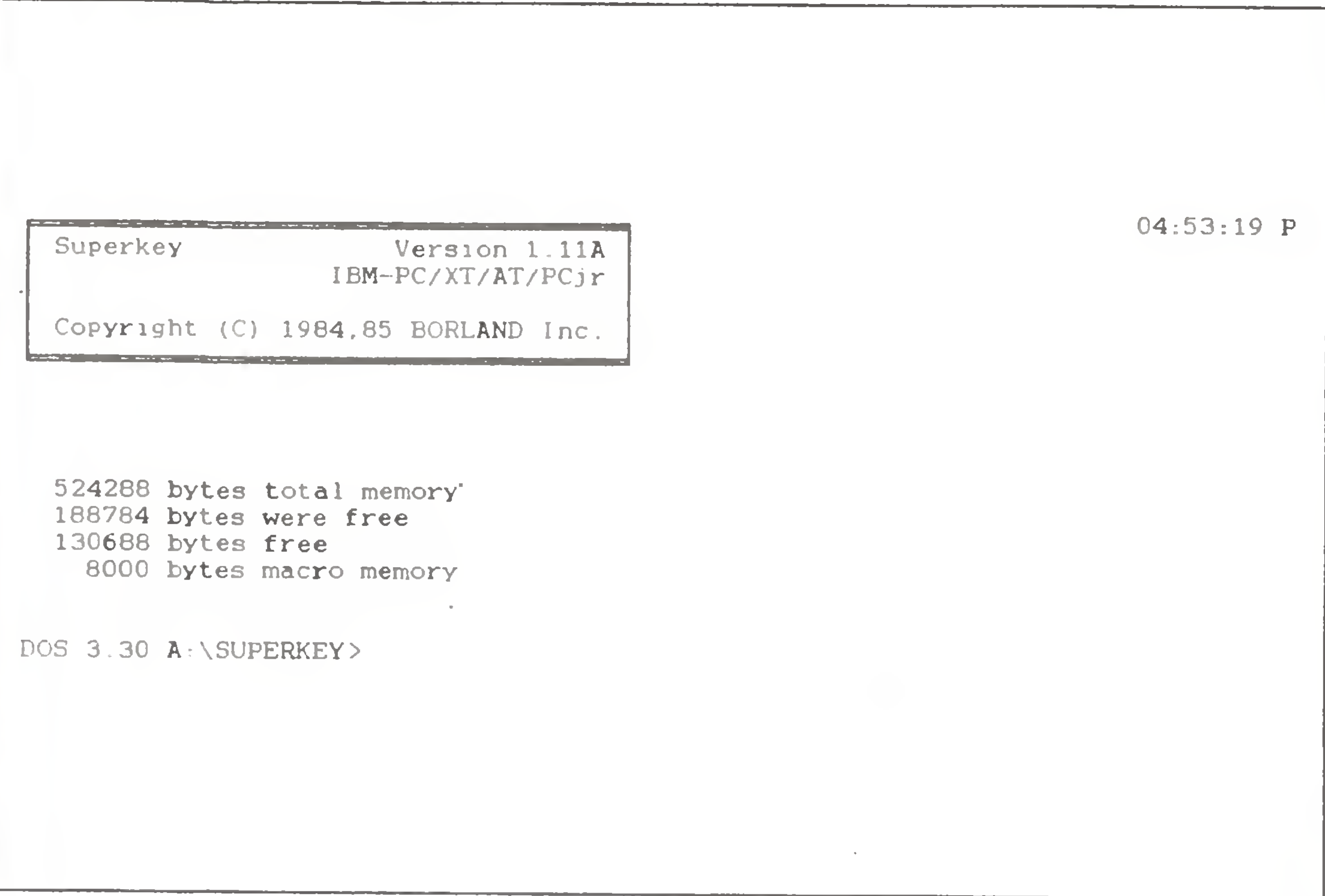
25 <
(który zajmuje ok.60KB i tworzy bufor dla klawiatury wielkości 128 znaków) oraz wielkość pamięci przeznaczonej na makrorozkazy. Ukazujący się nowy *prompt* świadczy, że nie opuściliśmy systemu. Aby dostać się do menu, trzeba nacisnąć <**Alt/**>. Jedynym sposobem usunięcia SuperKey z pamięci jest, podobnie jak w SideKicku, wciśnięcie przy głównym menu <**Ctrl**><**Home**><**End**>, trzeba jednak przy tym pamiętać, by **nie było** innych programów rezydentnych załadowanych po naszym programie.

Główne menu zawiera następujące opcje: **Macros, Commands Functions, Options, Defaults, Encryption, Layout i Setup.** Wyboru opcji dokonujemy **kursorem** i <**Enter**> lub wciskając wyróżnioną dużą literę w nazwie. Ukazują się wtedy menu rozwijane, w których postępujemy podobnie. W czasie pracy podpowiedź zawsze otrzymamy po wciśnięciu <**F1**>. Przyjrzyjmy się opcjom menu, zostawiając opis tworzenia makrorozkazów na koniec.

Setup:

Encryption directory: C:\SZYFRY – katalog plików o zaszyfrowanym dostępie;
Macro directory: C:\KEY\MAC – katalog plików makrorozkazów;
Save setup – zapamiętanie powyższych opcji;
SuperKey Version 1.15A - informacja o wersji programu.

Wywołując poszczególne opcje możemy poinstruować program, w jakich katalogach ma przechowywać pliki **.LAY** i pliki



.MAC oraz kazać te informacje zapamiętać, jeśli chcemy, by były one zawsze używane w chwili uruchamiania SuperKey.

Layout:

Load - załadowanie wskazanego pliku **.LAY;**
Clear - usunięcie aktywnego pliku **.LAY** z pamięci.

Utworzenie pliku .LAY następuje w wyniku uruchomienia programu **LAYOUT.COM**, który dokładnie objaśnia, jak to robić oraz wyświetla na ekranie klawiaturę oryginalną i przedefiniowaną.

Encryption:

Encrypt file - zaszyfrowanie dostępu do zbioru;
Decrypt file - odszyfrowanie dostępu do zbioru.

Po wywołaniu **Encrypt** pojawia się kolejne menu z pytaniami o nazwę pliku i sposób jego zapisania po zaszyfrowaniu dostępu i wreszcie o hasło. **Decrypt** oczywiście odwraca ten proces. Są dwa sposoby zapisania zaszyfrowanego pliku: tryb tekstowy, który pozwala na przesyłanie pliku przez łącze komuniacyjne i tryb nietekstowy. W obu trybach plik zaszyfrowany zostaje zapisany dokładnie w miejscu pierwotnej wersji. SuperKey **(KEY.COM)** używa szybkiego algorytmu, względnie łatwego do rozszyfrowania. Natomiast **KEYDES.COM** stosuje wolniejszy, ale o wiele sprytniejszy sposób. Należy przy obu metodach zalecić najwyższą ostrożność i sprawdzenie możliwości własnej pamięci, gdyż zapomnienie hasła może zakończyć się utratą danych. Nie wolno także mieszać metod szyfrowania i deszyfracji. Przechodzimy teraz do menu związanych z makrorozkazami:

Macros:

Begin - początek tworzenia makrorozkazu.
Title - nadanie etykiety makrorozkazowi do wyświetlenia przez **Commands shoW titles.**
Fixed Field - makrorozkaz będzie oczekiwać na podanie z klawiatury ciągu znaków o długości ustalonej <**Spacją**> lub symbolami formatującymi (**N** - cyfry, **L** - równany do lewego, **R** - do prawego marginesu, **C** - centrowany).
Var Field - makrorozkaz będzie oczekiwał na podanie z klawiatury ciągu znaków o dowolnej długości.
autO-start - automatyczne wykonanie makrorozkazu w chwili załadowania do pamięci pliku .MAC.
End - koniec tworzenia makrorozkazu.
cUt and paste - skopiowanie i zapamiętanie "pod" wybranym klawiszem fragmentu ekranu, przez oznaczenie górnego lewego rogu **kursorem** i <**B**>, a dolnego prawego **kursorem** i <**Enter**>. Fragment można wkopiować w inne miejsce po wciśnięciu wybranego klawisza.
eDit - włączenie edytora makrorozkazów.
Load - załadowanie do pamięci pliku .MAC.
Merge - równoczesne użycie kilku plików .MAC.
Clear - usunięcie makrorozkazów z pamięci.
Save - zapisanie na dysku makrorozkazów w pliku **.MAC.**

Functions - funkcje, które mogą być użyte tylko z makrorozkazu:

dAte - wypisanie daty z kalendarza systemu na ekranie.
Beep - sygnał dźwiękowy o podanym czasie trwania, wysokości początkowej i końcowej.
Clear screen - czyszczenie ekranu.
Delay factor - opóźnia wykonywanie makrorozkazu o podaną ilość odwołań programu użytkowego do klawiatury. Potrzebne przy wyprowadzaniu tekstów z makrorozkazów w niektórych wolno pracujących programach.
Keyboard lock - wył/włączenie klawiatury.
Logged drive - wyświetlenie oznaczenia bieżącego napędu dyskowego.
Path - wyświetlenie bieżącego katalogu i ścieżki.
Real time delay - jak Delay factor, ale czas mierzony w setnych sekundy.
Screen off - wyłączenie ekranu.
screen On - włączenie ekranu.
Time - wyświetlenie czasu z zegara systemowego.
cUt and paste - podobne do Macros-cUt and paste, ale bez zapamiętania fragmentu ekranu. Skopiowanie następuje natychmiast po wciśnięciu <**Enter**> w miejsce oznaczone kursorem przed rozpoczęciem realizowania funkcji.

Commands:

Command stack - wyświetlenie okna z komendami odłożonymi na stosie (do 256 znaków) i umożliwienie wybrania dowolnej z nich kursorem.
sKip macro - umożliwia wykorzystanie standardowych funkcji klawiszy przedefiniowanych w makrorozkazie.
stoP - przerywa działanie makrorozkazu, przydatne zwłaszcza przy beznadziejnych pętlach.
Sidekick - wywołanie SideKicka z makrorozkazu.
shoW titles - wyświetlenie okna ze zdefiniowanymi makrorozkazami i dotyczącymi ich tytułami (jeśli takowe zostały im nadane).

Options:

Arrow keys OFF - wył/włącza używane alternatywnie zamiast strzałek klawisze F7-F10. Można także wymienić te klawisze na inne.
Bottom line OFF - linia informacji w czasie tworzenia makrorozkazu wyświetlana na górze lub dole (ON) ekranu.
Command stack ON - wył/włącza odkładanie komend

```
Macros   Commands  Functions   Options   Defaults  Encryption Layout  Setup
                                         Begin macro     Alt=
                                         Title           Alt'
                                         autO-start      Alt`
                                         End macro       Alt-
                                         Menu            Alt/
                                         Fixed field     Ctrl]
                                         Var field       Ctrl-
                                         cUt and paste   Alt]
                                         beGin block     B
                                         sKip macro      `
                                         stoP            CtrlESC
                                         shoW titles     AltPRT
                                         Command stack   Alt\
                                         Decimal point   .
                                         deLimiter       ,
                                         fill cHaracter  <SPACE>
                                         help sYstem     F1
                                         Arrow keys
                                         Save defaults
```

		wyprowadzonych z klawiatury na stos (działa także z dBase).
Format fields	OFF	- wył/włącza formatowanie pól w obrębie makrorozkazu.
Keyb. click	OFF	- wył/włącza sygnał dźwiękowy towarzyszący naciskaniu klawiszy.
One finger	OFF	- ON pozwala na sekwencyjne naciskanie klawiszy w kombinacjach wymagających normalnie wciskania równoczesnego.
Playback delay	0	- opóźnia wykonywanie makrorozkazu o podaną liczbę przerwań sprawdzających stan klawiatury. Potrzebne przy wyprowadzaniu tekstów z makrorozkazów w niektórych wolno pracujących programach.
proTect delay	3	- czas w minutach, po którym wyłączany jest ekran, jeśli nie nastąpiła żadna akcja z klawiatury. Włączenie następuje po wciśnięciu dowolnego klawisza.
sUspend	OFF	- ON zawiesza działanie SuperKey. Uaktywnienie następuje po komendzie z systemu operacyjnego: **C:\KEY>KEY\OU.**
disk Wait	OFF	- ON zatrzymuje w buforze wszystkie kody podane z klawiatury w czasie pracy napędu dyskowego. Potrzebne w pracy z programami, które tego nie robią lub automatycznie opróżniają bufor.
Save options		- zapamiętanie opcji.

Defaults: zawiera kombinacje klawiszy wywołujących poszczególne funkcje SuperKey bez posługiwania się menu. Mogą być one dowolnie zmieniane i zapamiętane, by nie kolidowały z używanymi przez nas programami. Pozostałe pozycje to:

Decimal point	.	- znak oddzielający część całkowitą od dziesiętnej w polu sformatowanym.
deLimiter	,	- znak oddzielający liczby w sformatowanym polu cyfrowym.
fill cHaracter <SPACE>		- kody wypełniające niezapisaną część sformatowanego pola o stałej długości.
Arrow keys		- zdefiniowanie alternatywnych klawiszy kursora.
Save defaults		- zapamiętanie wszystkich ustaleń.

Tworzenie makrorozkazów jest bardzo proste, a dzięki zapisywaniu ich w plikach tekstowych może być dokonane na kilka sposobów. Pierwszy z nich polega na korzystaniu z menu lub z odpowiadających menu klawiszy podanych w **Defaults**. Drugi sposób, którego nie będziemy omawiać, gdyż świetnie tłumaczy się sam, opiera się na uruchomieniu funkcji **eDit** z menu **Macros**. Trzeci wreszcie, godny polecenia sprawnym użytkownikom, to redagowanie makrorozkazów w edytorze tekstów (nie wstawiającym znaków kontrolnych).

Spróbujmy utworzyć prosty makrorozkaz ułatwiający ustawianie parametrów pracy drukarki za pomocą komendy systemowej **MODE**, uruchamiany wciśnięciem **<AltP>**. Będziemy podawać literami dużymi klawisze wywołujące opcje z opisanych wyżej menu, małymi - teksty wyprowadzane z klawiatury oraz odpowiednie klawisze funkcyjne lub kombinacje klawiszy

<Alt/>MB<AltP>mode lpt1:<Alt/>MVN<Ctrl→,<Alt/>MFN<Ctrl]><Enter><Alt/>MTmode lpt1:X,Y<Enter><Alt/>ME

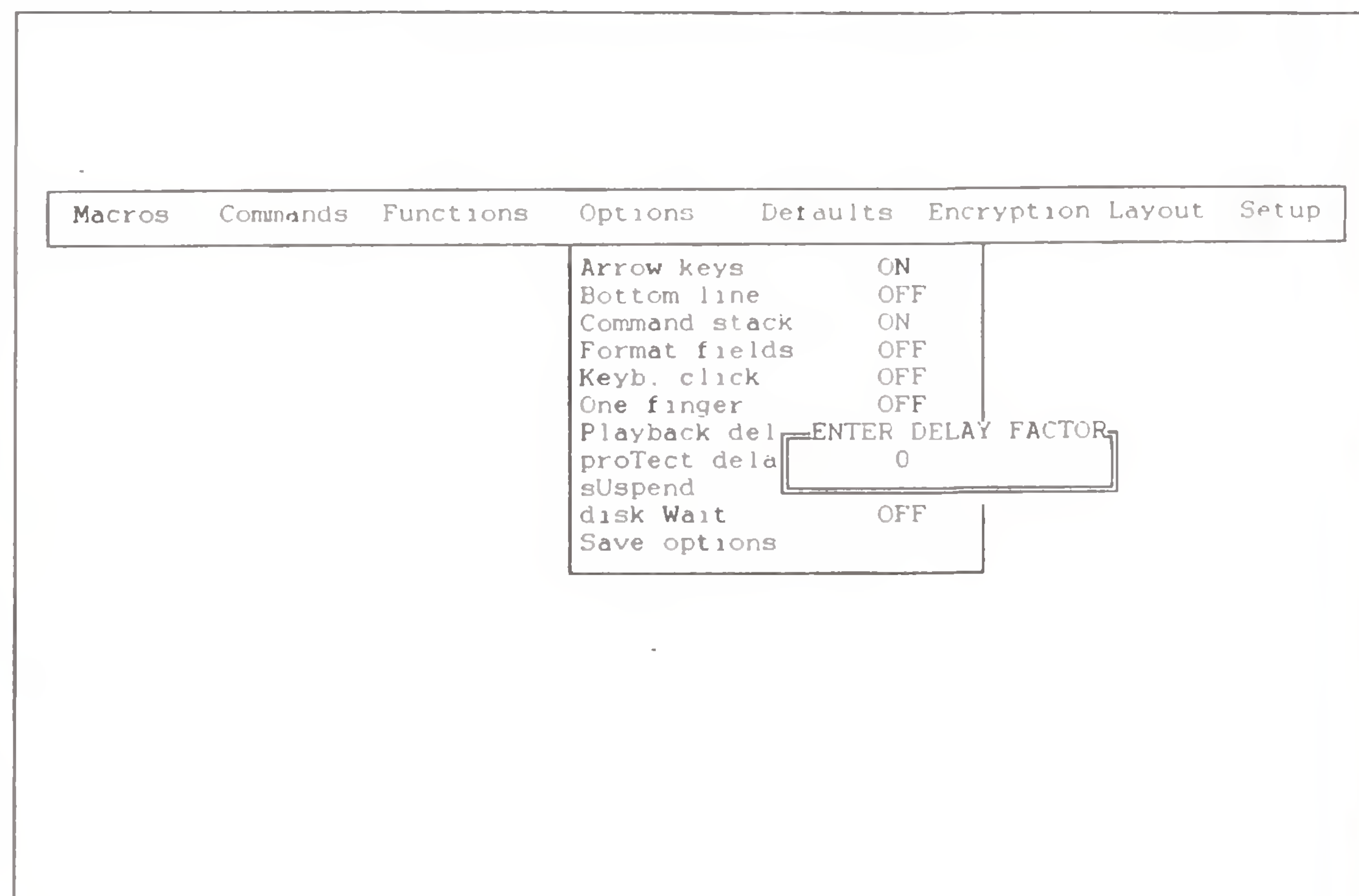
Objaśnienie: **<Alt/>** to wywołanie głównego menu, następnie kolejnych jego opcji**(Macros-Begin); <AltP>** - podanie w odpowiedzi na pytanie programu kombinacji zapamiętującej makrorozkaz; dalej treść komendy systemowej, ponowne wywołanie menu dla zdefiniowania pola o zmiennej długości **(Macros-Var Field)** o formacie przyjmującym tylko cyfry **(N - Numeric format)**; **<Ctrl->** kończy definicję pola; dalej, po przecinku, znów wracamy do menu dla zdefiniowania pola o stałej długości **(Macros-Fixed Field)** jednego znaku i formacie cyfrowym; **<Ctrl]** - koniec definicji pola; dodajemy jeszcze **<Enter>** aby rozkaz wykonany został automatycznie po wprowadzeniu żądanych wartości; kolejne powroty do menu dla nadania etykiety **(Macros-Title)** i zakończenia tworzenia **(Macros-End)**. W ten sam sposób można w makrorozkazy włączać opcje ze wszystkich menu z wyjątkiem **Encryption i Setup**. Od tej chwili makrorozkaz ten pozostaje w pamięci, natomiast dla zachowania go do dalszej pracy trzeba użyć opcji **Macros- Save**. Zapisany na dysku w pliku **.MAC** makrorozkaz będzie miał następującą formę:

<BEGDEF> <AltP><TITLE>mode lpt1:X,Y<TITLE>mode lpt1:<VFLD>N<VFLD>,<FFLD>N<FFLD><ENTER> <ENDDEF>

Dla piszących makrorozkazy w edytorach istotne są tu dwie informacje - jeśli w skład makrorozkazu wchodzi **<ENTER>**, to musi on kończyć w zapisie linię; każdy makrorozkaz musi zaczynać się od **<BEGDEF>** i kończyć **<ENDDEF>**. Wyjątek stanowią tu tzw. *display macros* czyli teksty do wyświetlenia na ekranie w specjalnym oknie (nie da się ich utworzyć bez edytora). Oto przykład:

<BEGDISP><AltF1>30,5,20,3.<TITLE>Ostrzezenie<TITLE> <AUTO><TRANSP>NIE DOTYKAC!<ENDDEF>

<AltF1> - ta kombinacja klawiszy wyświetli nasz napis; liczby kolejno: lewa kolumna, górna linia, szerokość, wysokość okna wraz z ramką; tytuł do wyświetlenia przez **shoW titles; <AUTO>** samoczynne wykonanie makrorozkazu po wczytaniu do pamięci zestawu, w skład którego on wchodzi; **<TRANSP>** - dalszą pracę można rozpocząć bez konieczności wciśnięcia **<Escape>** dla usunięcia napisu z ekranu. Włączenie w odpowiednie miejsca tekstu **<CtrlL>** podzieli go na strony, a **<CtrlB>** i **<CtrlD>** zmieni kolory tekstu oraz ramki w stosunku do wartości zadeklarowanych przy instalacji programu.



Użycie edytora tekstowego jest także konieczne, jeśli zamierzamy włączyć do makrorozkazów rozszerzony kod ASCII, gdyż przy aktywnym SuperKey znaki powyżej 126 są niedostępne z klawiatury.

Makrorozkazy mogą być również zagnieżdżone, co jest godne polecenia szczególnie, gdy wprowadzamy z klawiatury np. nazwę pliku, na którym dokonywać będziemy kilku operacji, a nie chcemy tej nazwy podawać za każdym razem. Najlepiej ten problem ilustruje przykład podręcznikowy:

<BEGDEF><AltC>copy <BEGDEF><CtrlF10> <ENDDEF> A:<ENTER> erase <CtrlF10><ENTER> <ENDDEF>

Sztuczka polega na tym, że w trakcie wykonywania głównego makrorozkazu jest tworzony nowy, który przyjmuje nazwę pliku wypisaną z klawiatury, a następnie zostaje wykonany po wyświetleniu na ekranie **ERASE**, oszczędzając nam ponownego podawania nazwy.

Doświadczenie uczy, że SuperKey nie współpracuje z wieloma programami działającymi w trybie graficznym (np. MS Word, Windows), popada też czasami w konflikt z programami rozszerzającymi klawiaturę (np. Fansi-Console), natomiast znakomicie się czuje w towarzystwie baz danych, edytorów, tabel itp. Przed użyciem programów graficznych lepiej wyrzucić SuperKey z pamięci, a co najmniej wyłączyć **proTect delay**, gdyż funkcja ta pozostaje aktywna i - przy pracy z Windows oraz innymi programami nie wyświetlającymi Help tekstowo - ekran gaśnie na dobre (zmuszając do wyjścia do systemu).

SuperKey może być wywoływany z plików wsadowych **(.BAT)**. Jako parametrów używać można wszystkich pozycji z **Options, Layout, Defaults** oraz **Macros: Load, Clear**. Działanie takie jest skuteczne zarówno przy pierwszym wywołaniu programu, jak i w czasie jego czuwania w pamięci. Na przykład: **C:KEY>key test/ML / OT+ /OW- /DV[Ctrl.]** spowoduje uruchomienie SuperKey z załadowanym zestawem makrorozkazów z pliku **TEST.MAC**, włączenie **proTect delay**, wyłączenie **disk Wait** i ustalenie, że pozycja **Var field** z **Macros** będzie wywoływana przez kombinację **<Ctrl.>**. SuperKey oddaje wielkie usługi w czasie pracy z programami o rozbudowanej "gałkologii" (np. LOTUS), oszczędzając wiele czasu i odciążając naszą pamięć. Szczególnie cenna wydaje się możliwość wprowadzania informacji z klawiatury w różnych zdefiniowanych uprzednio formatach. Miłą też jego cechą są ostrzeżenia i upomnienia zabezpieczające nas przed utratą makrorozkazów lub przedefiniowaniem uprzednio zdefiniowanego klawisza. ■



Adam Nowicki

Praktyka programowania

(Projekt programu)

Publikowany dziś odcinek cyklu "Praktyka programowania" w intencji autora oraz zgodnie z logiką pracy nad programem miał ukazać się jako rozpoczynający cykl. W wyniku naszego błędu pojawia się jako ostatni - poprzednie ukazały się w numerach 6/88 ("kodowanie") i 8/88 ("uruchamianie programu").

Niedawno widziałem program do obliczeń inżynierskich, który autorzy uznali za profesjonalny. Pod względem merytorycznym był bez zarzutu - stosował bardzo efektywny algorytm i dokładny model opisujący analizowany obiekt. Niestety, posługiwanie się nim wymagało ciągłego wertowania instrukcji obsługi. Wprowadzane dane były wyświetlane dopiero po zakończeniu obliczeń - przedtem nie wiadomo, czy je trzeba poprawiać, czy nie. Drukowany był tylko wynik końcowy, bez rezultatów pośrednich pozwalających doświadczonemu inżynierowi kontrolować poprawność obliczeń.

Pierwszą czynnością przed napisaniem dla kogoś programu musi być przełamanie bariery psychicznej. My wiemy, że nasze programy są bezbłędne, znamy na pamięć ich wszystkie opcje, wnętrze komputera rozgryźliśmy lepiej niż Piotruś Norton. Użytkownik naszego dzieła jest w znacznie gorszej sytuacji - należy mu trochę pomóc.

Mało kto wie dokładnie, czego oczekuje od programu. Przed jego napisaniem należy drobiazgowo omówić z przyszłym użytkownikiem wszystkie szczegóły: opcje, formaty wydruków, rodzaje wykorzystywanych drukarek, rozmiary przechowywanych danych i tym podobne. Najlepiej spisać to wszystko na papierze - pisanie zmusza do przemyślenia wielu spraw. Nawet jeśli piszemy program tylko dla siebie, warto przedtem sporządzić jego opis.

Kolejną sprawą jest dobór właściwego algorytmu. Duże programy używają wielu różnych algorytmów; wszystkie powinny być optymalne. Na przykład przy tworzeniu bazy danych musimy rozwiązać sortowanie, przeszukiwanie, wprowadzanie i wyprowadzanie informacji, ewentualnie także kompresję plików przed zapisaniem w pamięci masowej. Nie musimy i nawet nie powinniśmy wymyślać algorytmów sami - można je znaleźć w podręcznikach z serii Biblioteka Inżynierii Programowania. Od pewnego czasu algorytmy są publikowane w dziale KMK. Musimy uwierzyć, że stworzone i przeanalizowane przez zawodowych informatyków algorytmy są lepsze od wszystkiego, co moglibyśmy sami wymyślić. Dostatecznie dużo miejsca na inwencję pozostaje w tych częściach programu, które są nietypowe.

Dobrze jest, jeśli programista zna się na zagadnieniach rozwiązywanych przez program. Często jednak twory wąskich specjalistów z jakiejś dziedziny przypominają labirynt zrozumiały tylko dla nich samych. Najlepiej, żeby specjalista ściśle współpracował z programistą - przede wszystkim w fazie projektowania programu. W tym czasie poprawienie błędu merytorycznego jest o wiele łatwiejsze niż na końcu pracy.

Kiedy zawali się dom, projektantowi grozi więzienie. Nikt go nie spyta, czy używał naszego programu, czy przeprowadzał obliczenia na piechotę. Nie dziwmy się więc, że użytkownik może nie mieć do programu zaufania i może obawiać się pracować z nim. Zgodnie z maksymą "jak Cię widzą..." na zaufanie olbrzymi wpływ mają procedury współpracy z użytkownikiem. Musimy przyjąć zasadę maksymalnego ułatwienia pracy innej osobie - zawsze znającej program gorzej niż my sami. W dużych programach moduły komunikacji zajmują większość objętości, co trzeba uznać za normalne.

Wszystkie dostępne opcje muszą być wyświetlone na ekranie, a nie ukryte w podręczniku obsługi. Jeżeli menu ma strukturę drzewa, nie powinno być więcej niż dwa-trzy poziomy wyborów. Na każdym z nich należy umożliwić łatwy powrót do menu głównego - najlepiej zawsze za pomocą tego samego klawisza. Często się zdarza, że jedna z opcji jest wykorzystywana znacznie częściej niż inne. Wtedy udostępnienie jej przez naciśnięcie klawisza ENTER bardzo ułatwia pracę.

Stosowanie grafiki okien, niekoniecznie równocześnie z myszą, stwarza użytkownikowi komfort pracy. Bardzo mi się podobają rozwiązania wzorowane na Norton Commander, w których opcje zmienia się za pomocą kursora, wybrana możliwość jest podkreślona (na przykład przez rozjaśnienie tła), a do potwierdzenia wyboru służy klawisz ENTER. W większości translatorów języków dla IBM PC nie przewidziano firmowo grafiki okien - Fortran został pozbawiony jakiejkolwiek grafiki. Jeśli jednak pracujemy w dużym zakładzie czy na wyższej uczelni, na pewno znajdziemy znajomego, który kiedyś stworzył wszelkie potrzebne procedury.

Oczywiste są takie cechy dobrego programu, jak automatyczna obsługa błędów, wyświetlanie wszystkich wprowadzanych danych i częste pytania o konieczność dokonania poprawek. Jeśli program został przeznaczony dla szerszego grona użytkowników, powinien pracować z kilkoma najczęściej spotykanymi rodzajami kart graficznych, drukarek, pamięci masowych. W każdej chwili winien oferować zwięzłą, treściwą pomoc dla użytkownika - "HELP". Obecnie często programiści specjalnie utrudniają zrozumienie sposobu korzystania z programu, by uniemożliwić jego wykorzystywanie bez podręcznika użytkownika czy kursu obsługi. Jest to w naszych warunkach jedyna skuteczna ochrona programu przed nielegalnym kopiowaniem. Z punktu widzenia informatyka i użytkownika sposób jest nie tyle nieelegancki, co utrudniający życie.

Często są używane "efekty specjalne" - graficzne i dźwiękowe. Ich stosowanie powinno być uzależnione od przeznaczenia programu. Bardzo pożądane są one w grach i programach edukacyjnych dla dzieci. Pisząc program edukacyjny dla osób starszych należy już zachować ostrożność, całkowicie hamując twórcze zapędy w programach użytkowych. Zawsze trzeba mieć przede wszystkim na uwadze główny cel programu - jemu wszystko powinno być podporządkowane.

Gdy już dokładnie wiemy, co program ma robić, musimy rozbić zagadnienie na kilka mniejszych problemów. Podziału dokonujemy ściśle według struktury logicznej algorytmu. Na przykład program wymiarujący konstrukcję można podzielić na części:

- wprowadzenie danych o materiałach,
- wprowadzenie charakterystyki geometrycznej projektowanego obiektu,
- wprowadzenie danych o obciążeniach,
- obliczenie sił wewnętrznych występujących w elementach konstrukcji,
- znalezienie takich wymiarów przekrojów poprzecznych elementów, aby wytrzymywały działanie wyliczonych sił wewnętrznych.

Każdą z tych czynności można rozłożyć na mniejsze. Postępujemy tak dopóty, aż uzyskamy zadanie podzielone na proste procedury. Powyższy sposób postępowania jest nazywany programowaniem zstępującym.

Optymalną długością procedury jest kilkadziesiąt wierszy kodu źródłowego. Ważne jest, aby procedura realizowała jedno, wyraźnie określone, zadanie. Van Tassel sugeruje, by opisać pracę wykonywaną przez procedurę na kartce. Jeśli nie można tego uczynić w jednym zdaniu, podprogram należy podzielić na mniejsze. Procedurę o zalecanej długości łatwo zapamiętać, zrozumieć, zakodować i przetestować w ciągu godziny.

Każda procedura - zarówno najniższego rzędu, jak też wyższych - powinna mieć jedno wejście i jedno wyjście. Należy unikać stosowania instrukcji GO TO bezwarunkowo; rozkazy w podprogramie muszą być kolejno wykonywane od jego początku do końca. Wszystkie zmienne używane tylko wewnątrz procedury powinny być definiowane jako lokalne - o ile stosowany język na to pozwala.

Pisząc program musimy umożliwić dokonywanie w nim zmian. Przypuśćmy, że nasz przykładowy program wymiaruje konstrukcje żelbetowe. Taki mieliśmy napisać, zrobiliśmy swoje i zainkasowaliśmy honorarium. Po pewnym czasie zleceniodawca stwierdził, iż za pomocą programu chciałby również projektować budynki stalowe. Musimy odpowiednio uzupełnić pierwszą i ostatnią część programu - pozostałe zostaną bez zmian. Podprogramy powinny być zapisane w taki sposób, by uzupełnień dało się dokonać w drodze napisania dodatkowych procedur, dołączonych do menu. Im mniej trzeba zmieniać w już istniejących procedurach, tym lepiej. Praktycznie każdy większy program jest wielokrotnie przerabiany dla zwiększenia jego możliwości. Zawsze się znajdzie jakaś mutacja problemu wymagająca dodatkowych wierszy kodu, na której pominięcie ZDECYDOWALIŚMY SIĘ (podkreślam - musi być to efekt świadomego wyboru, a nie przypadku) początkowo.

Wybór języka jest często sprawą drugorzędną. Pracujący ze Spectrum i magnetofonem są skazani na Basic uzupełniany o procedury w asemblerze - im więcej funkcji systemu operacyjnego ma spełniać program i im szybciej działać, tym częściej trzeba używać

języka maszynowego. Realne jest tworzenie i przetestowanie dziennie dwóch kilobajtów Basica lub kilkuset bajtów kodu maszynowego. Dostateczną szybkość pracy programu zapewnia jego skompilowanie - najczęściej jest używany TOBOS FP. Napisanie dużego programu w Pascalu jest możliwe, ale bardzo żmudne. Stacja dysków pozwala szybciej pisać programy w asemblerze (jeśli popełniamy sporo błędów i często musimy wczytywać kompilator od nowa), ale nie w Basicu i Pascalu.

Istnieją na Spectrum wersje translatorów języków Forth i Logo, jednak nie widziałem dotąd programu użytkowego napisanego w którymś z nich.

Znacznie większy wybór mają użytkownicy IBM PC. Różne wersje Pascala i Fortranu, C, Basica, asembler... Basic jest używany do obliczeń podręcznych, ale napisanie krótkiego programu w Turbo Pascalu zajmuje tyle samo czasu. C jest uważany za język pozwalający modyfikować system operacyjny, jednak wiele na tym polu możemy zdziałać (na przykład wykorzystywać wszystkie funkcje MS-DOS i BIOS) używając Turbo Pascala. Zdecydowanie się na Fortran umożliwia wykorzystanie, po niewielkich przeróbkach, bogatej biblioteki programów dla maszyn Odra i Mera.

Decyzja o wyborze języka ma znaczenie przy pracy zespołowej lub wtedy, gdy program ma być w przyszłości czytany i modyfikowany. Większość firm tworzących oprogramowanie, w tym Microsoft, używa C i asemblera. Jeśli mamy napisać program samodzielnie, stosujmy taki język, by zakończyć pracę z minimalną liczbą błędów.

Jaki by nie był nasz wybór, przed rozpoczęciem pracy warto przejrzeć raport języka lub jakiś podręcznik używanej wersji. Kiedyś, gdy z Basica przesiadałem się na Pascala, wiele czasu straciłem wpisując SQR zamiast SQRT - dopiero rzut oka do podręcznika dopomógł w znalezieniu błędu. Na ogół warto zadać sobie trud dotarcia do oryginalnej dokumentacji - jest kompletniejsza niż tłumaczenia. Możemy też znacznie skrócić czas pracy używając gotowych i przetestowanych procedur bibliotecznych.

Literatura:
Dennie van Tassel "Praktyka programowania", WNT 1982



W pracy

Wiktor B. Daszczuk

Programowanie współbieżne [2]

Poprzednio ("Komputer" 4/88) omówiliśmy programowanie współbieżne z aktywnym oczekiwaniem. Jest to metoda rozrzutna: sprawdzając, czy spełniony został jakiś warunek, programy trwonią czas komputera, zwłaszcza gdy w kolejce czeka ich wiele. Program aktywnie oczekujący działa jak właściciel samochodu, co chwila wyglądający przez okno, zamiast kupić auto-alarm.

Taki auto-alarm stworzył profesor Dijkstra, a zwany jest on semaforem z nieaktywną formą oczekiwania. Semafor jest zmienną globalną, na której można wykonywać dwie operacje tradycyjnie oznaczane jako **P** i **V** (od holenderskich słów *passeren* - przejść i *vrygeven* - zwolnić).

Operacja **P** zmniejsza wartości dodatnie semafora o jeden lub zawiesza wykonujący ją program, póki semafor nie uzyska wartości dodatniej. Operacja **V** zwiększa wartość semafora o jeden, umożliwiając wykonanie na tym semaforze jednej z opóźnionych operacji **P**, o ile takie występują.

Przykład 1

Oto nowa wersja algorytmu produkcji i kradzieży śliwek (producenta i konsumenta, czyli buforowania) z nieograniczoną pojemnością bufora. Użyjemy dwóch semaforów. Jeden z nich **(-wypełnienie-)** pilnuje, by konsument nie pobrał więcej elementów, niż zawiera ich **BUFOR**. Drugi semafor **(-bezpieczeństwo-)** umożliwia wykluczanie się programów przy dostępie do bufora. Oto algorytm dla obu programów:

PRODUCENT:
1. **produkuj element**
2. **P(bezpieczeństwo)**
3. **umieść element w BUFORZE**
4. **V(bezpieczeństwo)**
5. **V(wypełnienie)**
6. **wróć do kroku 1**

KONSUMENT:
1. **P(wypełnienie)**
2. **P(bezpieczeństwo)**
3. **pobierz element z BUFORA**
4. **V(bezpieczeństwo)**
5. **konsumuj element**
6. **wróć do kroku 1**

Prawda, że jest on dużo prostszy od poprzedniego? Jest też oszczędniejszy, gdyż programy nie oczekują aktywnie. Początkowo BUFOR jest pusty, a więc **-wypełnienie-**=0 i żaden program z niego nie korzysta, więc **-bezpieczeństwo-**=1.

Czytelnikom proponuję zbudowanie algorytmu dla BUFORA ze skończoną pojemnością. Potrzebny będzie semafor symetryczny do semafora **-wypełnienie-** wskazujący liczbę wolnych miejsc w BUFORZE (początkowo równą jego pojemności).

Zakleszczenie

Co się stanie, gdy program kolejno dwukrotnie wykona operację **V** na semaforze typu "bezpiecznik"? Zapewne "wpuści" dwa inne programy do regionu krytycznego, a każdy z nich będzie ufał, że ma wyłączny dostęp do zmiennej. A w przypadku operacji **P**? Przy drugim jej wykonaniu semafor ma wartość 0 i operacja zostanie opóźniona. Kiedy program zostanie wznowiony? Nigdy, bo inny program musiałby opuścić region krytyczny, a wcześniej doń wejść, co uniemożliwia wyzerowany semafor! Stan, w którym program czeka na spełnienie niewykonalnego warunku, nazywamy zakleszczeniem (ang. *deadlock*, spotyka się też inne polskie tłumaczenia tego terminu). Problem unikania go przekracza ramy tego artykułu.

Monitor

Koncepcja semafora z nieaktywnym oczekiwaniem była przełomem. Nowe rozwiązania pojawiły się potem lawinowo. Omówimy jedno z nich, związane blisko z pojęciem klasy (omawiałem je w "Komputerze" 4/87).

Przypomnijmy, że klasa to zespół zmiennych oraz procedur, których wywołanie jest jedynym sposobem dostępu do zmiennych klasy. Klasa jako mechanizm programowania współbieżnego musi być dostępna dla wszystkich programów (jak semafor) i wywołania jej procedur muszą się wykluczać. Możliwe musi być też zawieszenie działania programu do czasu spełnienia jakiegoś warunku. W semaforze warunek taki jest badany wraz z zajęciem regionu krytycznego. Wywołanie procedury klasy wyklucza co prawda wywołanie jej procedur przez inne programy, ale gdy operacja na zmiennej klasy nie może być wykonana (na przykład nie można pobrać elementu z pustego bufora), mamy kłopot. Można zakończyć procedurę i wywołać ją ponownie (może tymczasem inny program zapełni bufor), ale jest to oczeki-

> 30

29 <

wanie aktywne, którego mamy unikać. Profesor Hoare zaproponował wprowadzenie zmiennych typu **-condition-** (warunek). Wykonanie operacji **-wait-** (czekaj) na takiej zmiennej zawiesza program wywołujący, póki inny program nie wykona operacji **-signal-** (sygnalizuj spełnienie warunku). Program oczekujący jest wznawiany bezpośrednio po tej operacji.

Klasę o wymienionych cechach nazywamy monitorem. Jest to drugie, obok semafora, podstawowe narzędzie programowania współbieżnego.

Przykład 2

Oto algorytm "producent - konsument", wykorzystujący monitor, tym razem dla bufora o ograniczonej pojemności:

```
monitor BUFOR:
  zmienne C i D typu WARUNEK
  procedura UMIEŚĆ
    1. jeżeli bufor jest pełny to C.wait
    2. umieść element w buforze
    3. D.signal
  procedura POBIERZ
    1. jeżeli bufor jest pusty to D.wait
    2. pobierz element z bufora
    3. C.signal
  program PRODUCENT:
    1. produkuj element
    2. BUFOR.UMIE{C
    3. wróć do kroku 1
  program KONSUMENT:
    1. BUFOR.POBIERZ
    2. konsumuj element
    3. wróć do kroku 1
```

Monitor a semafor

Monitor ma dwie przewagi nad semaforem: operacje na zmiennej globalnej są zgrupowane, a zajmowanie i zwalnianie regionu krytycznego jest niepodzielną operacją wywołania monitora (na przykład BUFOR.POBIERZ). Wyklucza to zapomnienie o operacji P lub V na którejś ze ścieżek programu lub wykonanie jej dwukrotnie, na przykład w pętli. Do synchronizacji można też użyć dowolnego warunku, nie tylko przejścia przez zero, jak w semaforze.

Są to jednak mechanizmy równoważne, tzn. można zbudować zarówno monitor z semaforów, jak i semafor za pomocą monitora.

Przykład: jądro systemu operacyjnego

Oto algorytm jednoprocesorowego jądra zapisany w języku Concurrent Pascal (Pascal rozbudowany o konstrukcje klasy i monitora). Podstawowym użytym typem danych jest opis procesu (rekord **-proces-**), zawierający dwa pola: pole -**rejestry**-pozwala oczekującenu procesorowi sądzić, że ma "własny" (wirtualny) procesor, którego stan rejestrów jest przechowywany w tym polu. Jego zawartość po uaktywnieniu procesu trafia do rejestrów procesora.

Pole **-następny-** jest wskaźnikiem na następny element w kolejce procesów (opisuje ją klasa **-kolejka-**). Głowa kolejki jest wskaźnikiem na pierwszy element, pierwszy element zawiera wskaźnik na drugi itd. Ostatni element (z początku jest nim głowa kolejki) zawiera wskaźnik **-nil-**: "donikąd". Procedura **-dołącz-** dołącza proces na koniec kolejki, a procedura **-weź-** pobiera z niej pierwszy proces, gdy kolejka nie jest **-pusta-**. Jest to tzw. **kolejka FIFO**, z której pobierany jest zawsze element najdłużej w niej przebywający. Nasze jądro działa więc "sprawiedliwie".

Klasa -**semafor**-zawiera zmienną semaforową -licznik- i zmienną **-zawieszone-**, będącą kolejką procesów, które wykonały operację **P**, gdy **-licznik-** był równy 0. Opis semafora zawiera procedury **-P-** i **-V-**. Jego wartość początkowa jest parametrem procedury inicjującej w nagłówku klasy.

Jądro (monitor **-jądro-**) korzysta z dwóch zmiennych. Są to: wskaźnik na proces aktywny oraz kolejka procesów gotowych do realizacji.

Procedury monitora **-jądro-** mogą być wywoływane przez procesy. Każdą z nich (oprócz inicjującej) rozpoczyna zapis stanu rejestrów procesora dla procesu aktywnego, a kończy się odtworzeniem ich wartości z opisu (nowego) procesu aktywnego. Operacji tych nie da się opisać w języku Concurrent Pascal, trzeba skorzystać z asemblera.

W procedurze **-P-**, gdy po zawieszeniu procesu (**-licznik-** =0) kolejka procesów gotowych jest pusta, a więc żaden proces nie może wykonać operacji V (być może wszystkie procesy są zakleszczone), jądro kończy działanie.

Procedura **-start-** uruchamia nowy proces. Proces wołający ją musi podać stan rejestrów nowego procesora wirtualnego.

Przełączanie procesów

Jeżeli jeden z procesów wykonuje (poprawnie lub w wyniku błędu) pętlę nieskończoną bez wywołań operacji P, to żaden inny proces nie będzie realizowany.

Zdefiniujmy więc procedurę **-tik-**, której wywołanie przez zegar zmienia proces aktywny, co daje procesom z kolejki dostęp do procesora w rytm przerwań zegara, np. co 20 ms. Opis jej wykracza poza Concurrent Pascal.

Przy takim rozwiązaniu pojawia się następny problem. Otóż zegar może przerwać jedną z procedur jądra i region krytyczny kolejki procesów nie będzie zachowany. Podczas wykonywania procedur jądra przyjmowanie przerwań musi być więc blokowane. Blokowania przerwań nie można opisać w języku Concurrent Pascal.

Jądro wieloprocesorowe

Jak zmodyfikować nasze jądro, by mogło funkcjonować w komputerze wieloprocesorowym? Trzeba określić wskaźniki procesów aktywnych dla wszystkich procesorów, np. deklarując tablicę wskaźników.

Po drugie, procesy z dwóch procesorów mogłyby teraz na przykład jednocześnie wywołać operację **V** na tym samym procesorze. Konieczna jest więc niepodzielna operacja "czytaj i pisz" i aktywne oczekiwanie. Tego fragmentu jądra także nie opiszemy w języku Concurrent Pascal.

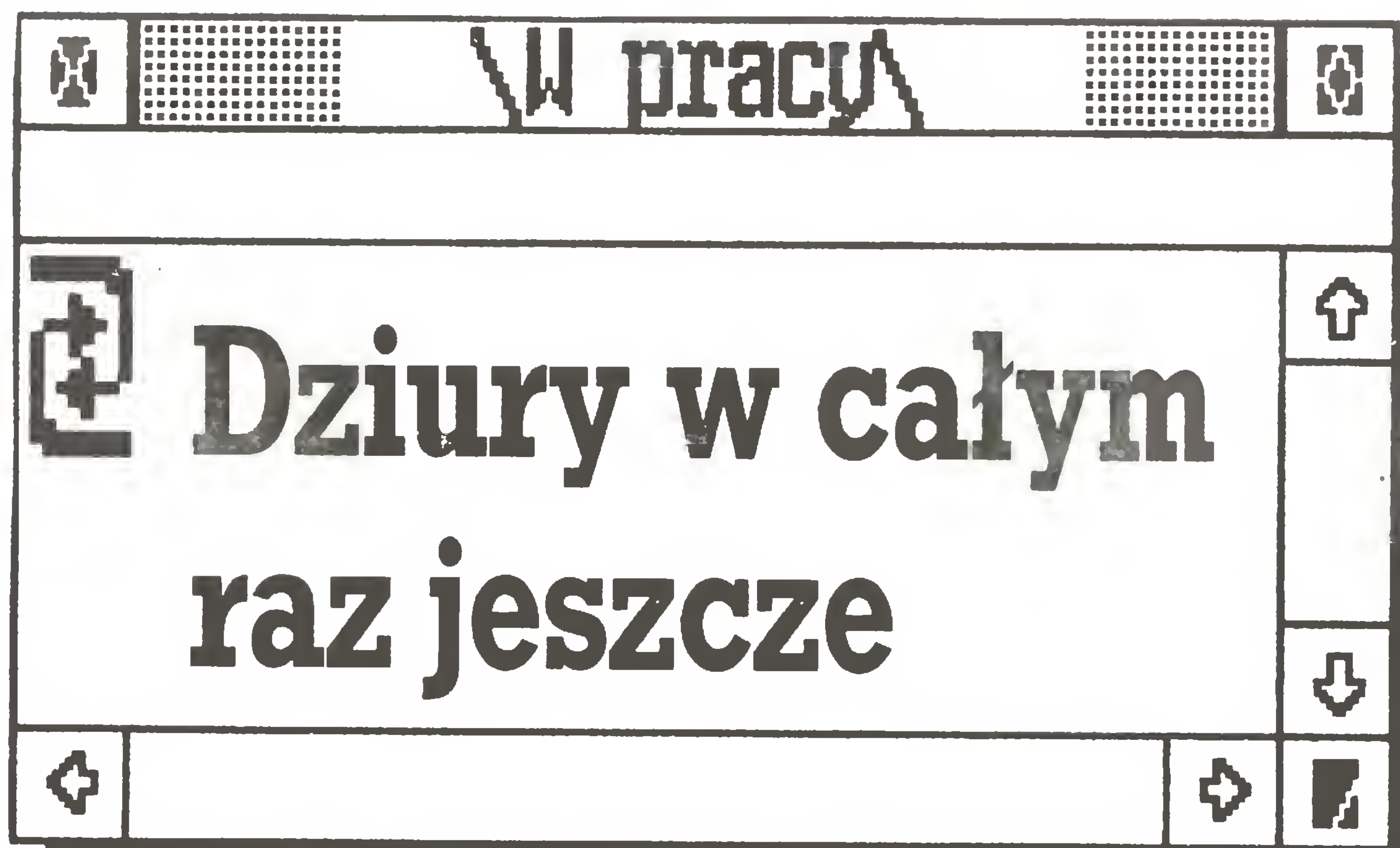
Po trzecie, pusta kolejka procesów gotowych przy zawieszaniu procesu procedurą **'P'** nie oznacza jeszcze zakleszczenia wszystkich procesów, gdyż mogą występować procesy aktywne na innych procesorach. Jądro może oczekiwać aktywnie na pojawienie się opisu procesu w kolejce lub zatrzymać procesor i czekać na przerwanie aktywujące. Trzeba je wysłać, gdy na innym procesorze zostanie wznowiony jakiś proces. W obu wypadkach jednak jądro musi zwolnić region krytyczny, aby na innym procesorze mogła być wywołana procedura **-V-**, która wznowi jeden z procesów. Instrukcja "zatrzymaj" nie może być opisana w języku Concurrent Pascal.

Na koniec proponuję Czytelnikowi takie zmodyfikowanie jądra, aby proces mógł wstrzymać swoje działanie o określony czas (liczbę przerwań od zegara czasu rzeczywistego).

Opisany algorytm jest bardzo uproszczony, pomija m.in. współpracę procesów z urządzeniami, ilustruje jednak metody rozwiązywania problemów współbieżności. Autor zrealizował bardzo podobne jądro dla programów w Fortranie i C na komputerze IMP-85 w systemie CP/M 2.2. Zajmuje ono mniej niż 1000 bajtów pamięci.

```
algorytm "jądro"
  type wiązanie = ^proces;
proces = record rejestry: array [...] of integer;
   następny: wiązanie
         end;
  kolejka = class;
    var głowa: wiązanie;
    procedure entry dołącz (pr: wiązanie);
      var x: wiązanie;
      begin
        pr^.następny := nil;
        if głowa = nil
          then głowa := pr
          else begin
            x := głowa;
            while not (x^.następny = nil)
              do x := x^.następny;
            x^.następny := pr
    end end;
  procedure entry weź (var pr: wiązanie);
    begin
      if głowa = nil
        then pr := nil
        else begin
          pr := głowa;
          głowa := głowa^.następny
    end end;
  function entry pusta: boolean;
    begin
      pusta := głowa = nil
    end;
  begin
    głowa := nil
  end; (* klasa kolejka *)
semafor = class (wartośćpoczątkowa: integer);
  var licznik: integer;
      zawieszone: kolejka;
  procedure entry P (aktywny: wiązanie;
                                   var zawieszono: boolean);
    begin
      if licznik > 0
        then begin
          licznik := licznik-1;
          zawieszono := false
        else begin
          zawieszone.dołącz (aktywny)
          zawieszono := true
    end end;
procedure entry V (var wznowiony: wiązanie);
    begin
      if zawieszone.pusta
        then begin
          licznik := licznik+1;
          zawieszony := nil
        else zawieszone.weź (wznowiony);
    end;
  begin licznik := wartośćpoczątkowa;
        init zawieszone
  end; (* semafor *)
var jądro: monitor (pr : wiązanie);
  var aktywny: wiązanie;
      gotowe: kolejka;
  procedure entry P (s: semafor);
    var zawieszono: boolean;
    begin
      zachowajrejestry (aktywny^.rejestry);
      s.P (aktywny, zawieszono);
      if zawieszono
        then if gotowe.pusta
          then (* koniec działania jądra *)
          else begin
            gotowe.weź (aktywny);
            odtwórzrejestry (aktywny^.rejestry)
    end end;
  procedure entry V (s: semafor);
```

```
var wznowiony: wiązanie;
  begin
    zachowajrejestry (aktywny^.rejestry);
    s.V (wznowiony);
    if not (wznowiony = nil)
      then begin
        gotowe.dołącz (wznowiony);
        odtwórzrejestry (aktywny^.rejestry)
  end end;
procedure entry inicjuj (s : semafor;
                          wartośćpoczątkowa: integer);
  begin
    zachowajrejestry (aktywny^.rejestry);
    init s (wartośćpoczątkowa);
    odtwórzrejestry (aktywny^.rejestry)
  end;
procedure entry start (pr: wiązanie)
  begin
    zachowajrejestry (aktywny^.rejestry);
    gotowe.dołącz (pr);
    odtwórzrejestry (aktywny^.rejestry)
  end;
procedure entry tik;
  begin
    zachowajrejestry (aktywny^.rejestry);
    if not gotowe.pusta
      then begin
        gotowe.dołącz (aktywny);
        gotowe.weź (aktywny)
      end;
    odtwórzrejestry (aktywny^.rejestry)
  end;
begin
  aktywny := pr;
  init gotowe;
  odtwórzrejestry (aktywny^.rejestry)
end (* jądro *)
```



Szanowny Panie Redaktorze,

z wielką przyjemnością przeczytałem interesujący artykuł Andrzeja Kadlofa "Dziury w całym" ("Komputer" 3/88)

Sprawdziłem działanie przykładowych programików zaprezentowanych w tym artykule. Próby potwierdziły wyniki uzyskane przez Andrzeja Kadlofa z dwoma wyjątkami.

Problem 1. Zapamiętanie zawartości pamięci ekranu na dysku i powtórny odczyt (użycie instrukcji Blockread i Blockwrite).

Wyniki uzyskane przez Kadlofa nie potwierdziły się w komputerze TC2000 firmy TRADECOM i komputerach EMIX; w przypadku sprzętu dostarczanego przez AGROCOMPUTER wynik zależał od egzemplarza. W komputerach TC2000 i EMIX odczyt i zapis zawartości pamięci ekranu do/z zbioru dyskowego przebiegał bezbłędnie. W celu wyeliminowania wpływu kompilatora na wykonanie programu zrealizowałem ten sam algorytm w języku GWBASIC. Wykonanie programu z wykorzystaniem interpretera GWBASIC dało takie same wyniki. Przyczyna nie leży zatem w błędzie kompilatora, ale raczej w nieprawidłowej współpracy między tym fragmentem pamięci operacyjnej (tzn. pamięci ekranu) i układem DMA (bezpośredniego dostępu do pamięci) oraz sterownikiem monitora. Jak wiadomo, obszar pamięci ekranu rozciągający się od adresu $B000:0000 znajduje się fizycznie na pakiecie (karcie) sterownika monitora Hercules. Układy wyświetlające muszą co 20 ms (1/50 s) pobrać zawartość pamięci celem odświeżenia treści ekranu monitora. Z drugiej strony każda operacja dyskowa jest wykonywana z wykorzystywaniem układu bezpośredniego dostępu do pamięci (ang. DMA). Z tego wynika, że może wystąpić konflikt w żądaniu dostępu do pamięci przez układy wyświetlania i przez układ DMA. W efekcie nieprawidłowego zarządzania dostępem do pamięci układ DMA "nie trafia" w bajty znakowe, tylko w bajty atrybutów znaków. Przypomnijmy, że organizacja pamięci ekranu w komputerach PC/XT wygląda tak, że każdemu bajtowi danych (znak do wyświetlenia) towarzyszy bajt atrybutu. Podczas transmisji dyskowej efekt "rozsynchronizowania" adresacji dla układu DMA występuje w obu kierunkach (tzn. zapis na dysk i odczyt z dysku).

Reasumując: obserwacja A. Kadlofa była trafna, ale wyciągnięte wnioski błędne. Problem nie jest natury programowej, ale sprzętowej.

Problem 2. Działanie procedury Delete.

Pozwoliłem sobie zanalizować działanie procedury Delete. Otóż wykonanie procedury Delete na zmiennej łańcuchowej (ang. *string*) powoduje następujące czynności:

a) skopiowanie w lewo znaków będących na prawo od kasowanego fragmentu.

b) zmodyfikowanie długości zmiennej.

Słowa "skopiowanie" a nie "przeniesienie" użyłem celowo, ponieważ zmienia się wprawdzie długość zmiennej tekstowej, niemniej obszar pamięci zajęty przez poprzednią długość jest wypełniony ostatnim znakiem. Najlepiej zilustruje to przykład: Niech zmienna łańcuchowa **s** zadeklarowana jako string [10] ma wartość '?****ALA'. Używając procedury Delete skasujemy pierwszy znak, tzn. użyjemy Delete(s,1,1).

Przed wykonaniem procedury Delete(s,1,1) obszar pamięci zarezerwowany na zmienną **s** wygląda następująco:

8 bin	?	*	*	*	*	A	L	A	x	x
0	1	2	3	4	5	6	7	8	9	10

x – wartość nieoznaczona

a po wykonaniu Delete(s,1,1) wygląda zaś tak:

7 bin	*	*	*	*	A	L	A	A	x	x
0	1	2	3	4	5	6	7	8	9	10

x – wartość nieoznaczona

Przykład ten ukazuje, dlaczego program napisany przez Andrzeja Kadlofa zawierający procedurę USUWAJ "zawiesza się", gdy jako argument tej procedury położymy ciąg samych gwiazdek (w ogólności - ciąg tych samych znaków). Otóż warunek s [1] = ch będzie zawsze prawdziwy - miejsce pamięci o adresie zawartym w wyrażeniu s [1] będzie zawierało gwiazdkę (obszar pamięci przewidziany na zmienną **s** będzie w całości wypełniony gwiazdkami - co wynika z zaprezentowanego wyżej przykładu). A zatem procedura usuwania wpada w nieskończoną pętlę.

Przyczyna takiego zachowania się procedury USUWAJ nie leży po stronie kompilatora, tylko, niestety, po stronie Autora, Andrzeja Kadlofa. Stosuje on bowiem wyrażenia tablicowe (a niewątpliwie s [1] jest wyrażeniem tablicowym) do operacji na zmiennych łańcuchowych. Autor próbuje ratować sytuację badając warunek (s [1.] = ch) and (s<>") ale - choć rozwiązanie jest skuteczne - to moim zdaniem nieeleganckie. Po operacji na zmiennych łańcuchowych (ang. *string*) powinno się używać przewidzianych do tego procedur. Proponuję tu użycie procedury COPY, która w podprogramie USUWAJ wyglądałaby następująco:

```
.....
while COPY(s,1,1)=ch do
.....
```

Proszę zwrócić uwagę, że warunek w pętli while będzie fałszywy, jeśli łańcuch **s** będzie pusty.

Mógłby ktoś zauważyć, że zmienne łańcuchowe to też tablice, tyle że zmiennej długości, więc i wyrażenia tablicowe też powinny być dozwolone. Tak, ale po to twórca kompilatora wprowadził typ string i operacje dozwolone na tym typie zmiennych (w postaci operatora łączenia zmiennych łańcuchowych - ang. *concat* - i procedur standardowych), żeby "przykryć" fizyczne zachowanie się zmiennej tego typu. Wynika z tego wniosek, że traktowanie zmiennej łańcuchowej jako tablicy jest oczywiście dozwolone, ale wymaga pełnej świadomości programisty co do ewentualnych skutków takiego postępowania. Ta uwaga nie jest niestety wyrażona explicite w oryginalnej dokumentacji Turbo Pascala wersja 3.00, niemniej trzeba zdawać sobie sprawę, że zmienna łańcuchowa jedynie przypomina zmienną tablicową (ang. *array*). Nawiasem mówiąc, dokumentacja do PASCALA MT+ ostrzega przed bezmyślnym traktowaniem zmiennych łańcuchowych jako zmiennych tablicowych.

Reasumując:

Przyczyna złego działania procedury USUWAJ nie leży po stronie kompilatora. Wydaje mi się, że twórca kompilatora. Turbo Pascal celowo "zdjął" ochronę ze sprawdzania zgodności stosowanych wyrażeń tablicowych w stosunku do zmiennych łańcuchowych. Miał prawdopodobnie na celu umożliwienie bardziej "wyrafinowanego" programowania. Z praktyki wiem, że np. procedura POS działa haniebnie wolno; w pewnym programie "uwarunkowanym czasowo" musiałem napisać własną, specjalizowaną procedurkę realizującą podobną funkcję. Korzystałem przy tym z wyrażeń tablicowych na zmiennych łańcuchowych.

Pozostałe efekty dostrzeżone przez Andrzeja Kadlofa powtarzają się we wszystkich komputerach.

Łączę wyrazy szacunku
Krzysztof Fietkiewicz
Warszawa

Stephen K. O'Brien w swojej książce ***Turbo Pascal: The Complete Reference*** *(str. 258) przedstawia procedurę StripBlanks praktycznie identyczną z moją (usprawnioną wersją Usuwaj). Można by zatem przyjąć, że niedouczonych programatorów Turbo Pascala jest na świecie co najmniej dwóch, gdyby nie fakt, że książka ta ukazała się w USA w roku 1988 nakładem Borland-Osborne/McGraw-Hill, jej autor jest czołowym programistą Borland International, a Philippe Kahn (szef firmy) napisał we wstępie, aby przytoczone w niej programy traktować jako wzorcowe przykłady poprawnego stosowania języka. Sądzę, że to ostatecznie wyjaśnia intencje twórców Turbo Pascala i ich zalecenia dotyczące stosowania procedury Delete. Jej implementację będę w dalszym ciągu uważał za co najmniej niedoskonałą.*

Andrzej Kadlof



Roland Wacławek

Ujarzmianie myszki [3]

W trybie graficznym można nadać kursorowi dowolny kształt, który da się skomponować w siatce 16x16 punktów. Program obsługi (ang. *handler*) myszki zawiera wzorzec standardowego kursora graficznego, którym jest strzałka. Do definiowania innych postaci kursora graficznego służy funkcja usługowa nr 9.

Definicja kursora odwołuje się do dwóch tablic złożonych z bitów i nazwanych odpowiednio wzorcem kursora i maską kursora. Każdy bit w tablicy jest związany z pojedynczym bitem rastra, tak więc każda tablica składa się z 16*16/8 = 32 bajtów, inaczej: 16 słów. Można zatem przyjąć, że każdy element tablicy opisuje pojedynczy wiersz kursora. Wzorzec wspólnie z maską decydują o tym, jaki będzie właściwy kształt kursora, tzn. który punkt jego matrycy stanie się elementem tła, który zaś - wypełnienia ekranu. Mówiąc ściślej: maska kursora poddawana jest operacji iloczynu logicznego (AND) z zawartością ekranu, natomiast bezpośrednio potem rezultat poprzedniej operacji na ekranie zostanie poddany operacji różnicy symetrycznej (EX-OR, w skrócie: XOR) z zawartością wzorca kursora. Tak więc o postaci kursora w każdym z 16x16 punktów matrycy decyduje równocześnie para bitów: 1 we wzorcu kursora i 1 w masce kursora. Stan punktu ekranu w zależności od stanu odpowiednich bitów wzorca i maski zawiera tabela 1.

Tabela 1.

Wzorzec kurs.	Maska kursora	Stan punktu na ekranie
0	0	Punkt przyjmie barwę tła
1	0	Punkt przyjmie barwę wypełnienia
0	1	Punkt zachowa stan poprzedni
1	1	Nastąpi inwersja punktu

Celownikiem nazywamy wyróżniony punkt matrycy kursora, którego współrzędne reprezentują współrzędne kursora graficznego. Domniemane położenie celownika to lewy górny punkt matrycy. W razie potrzeby można jednak uczynić celownikiem dowolny inny punkt matrycy, podając odpowiednie parametry programowi obsługi myszki. Byłoby to konieczne np. wtedy, gdyby kursor miał przyjąć postać krzyża. W tym przypadku celownikiem powinien zostać centralny punkt matrycy.

Aby zdefiniować nową postać kursora graficznego, należy przekazać programowi obsługi w rejestrach BX i CX współrzędne X i Y celownika (dozwolone są wartości 0..15), natomiast w parze rejestrów ES:DX - absolutny adres nowej definicji kursora. Definicja składa się z 32 słów, z których pierwsze 16 tworzy tablicę maski, pozostałe - tablicę wzorca kursora. W Turbo Pascalu każdą z tablic możemy zdefiniować następująco:

```
TYPE Raster_kursora = ARRAY[0..15] OF Integer;
```

Przypisywanie kursorowi nowych form może ułatwić poniższa procedura. Dwa pierwsze parametry to współrzędne celownika w stosunku do lewego górnego rogu rastra kursora. Dwa następne - to tablice typu Raster_kursora. Możliwość odrębnego podawania wzorca i maski może być niekiedy bardzo przydatna:

```
PROCEDURE Definiuj_kursor_graf(Xcel, Ycel: Integer;
                          Maska, Wzorzec: Raster_kursora);
  VAR Rejestry: Rejestry_8088;
    Def_kursora: ARRAY[1..2] OF Raster_kursora;
BEGIN
  Def_kursora[1]:= Maska;
  Def_kursora[2]:= Wzorzec;
  Rejestry.AX:= 9; Rejestry.BX:= Xcel; Rejestry.CX:= Ycel;
  Rejestry.DX:= Ofs(Def_kursora);
  Rejestry.ES:= Seg(Def_kursora);
  Intr(51, Rejestry)
END;
```

Przypuśćmy, że chcemy zdefiniować kursor w formie strzałki, np. takiej oto:

```
. . . . . . . . . . . . . . . .
. # . . . . . . . . . . . . . .
. # # # . . . . . . . . . . . .
. # # # # # . . . . . . . . . .
. # # # # # # # . . . . . . . .
. # # # # # # # # # . . . . . .
. # # # # # # # # # # . . . . .
. # # . . . # # . . . . . . . .
. . . . . . . # # . . . . . . .
. . . . . . . . # # . . . . . .
. . . . . . . . . # # . . . . .
. . . . . . . . . . # # . . . .
. . . . . . . . . . . # # . . .
. . . . . . . . . . . . # # . .
. . . . . . . . . . . . . . . .
```

Jeśli strzałka będzie przemieszczana po pustym ekranie, to sprawa jej widoczności nie budzi wątpliwości. A jeśli ekran będzie częściowo lub całkowicie wypełniony? Jeżeli maskę kursora złożymy z samych zer, kursor będzie występować w prostokątnej, pustej ramce. Jeżeli maskę kursora stanowić będą wyłącznie jedynki, kursor będzie przedstawiany inwersyjnie, tzn. tam, gdzie ekran jest jasny, kursor będzie ciemny i odwrotnie. W praktyce najkorzystniej jest nadać wnętrzu kursora jasną barwę, natomiast jego kontur spreparować tak, by na jasnym tle był zawsze ciemny. W ten sposób na jasnym tle kursor będzie wyraźnie odcinał się ciemnym konturem. Jak to osiągnąć? Oto przykładowa maska dla naszej strzałki:

```
. . # # # # # # # # # # # # # #
. . . . # # # # # # # # # # # #
. . . . . . # # # # # # # # # #
. . . . . . . . # # # # # # # #
. . . . . . . . . . # # # # # #
. . . . . . . . . . . . # # # #
. . . . . . . . . # # # # # # #
. . . . . . . . # # # # # # # #
. . . # # . . . . # # # # # # #
# # # # # # . . . . # # # # # #
# # # # # # # . . . . # # # # #
# # # # # # # # . . . . # # # #
# # # # # # # # # . . . . # # #
# # # # # # # # # # . . . . # #
# # # # # # # # # # # . . . . #
# # # # # # # # # # # # # . . #
```

Jak widać, w masce wyzerowane jest nie tylko wnętrze strzałki, ale i łańcuch punktów w bezpośrednim sąsiedztwie jej konturu. Tam, gdzie maska jest wyzerowana, kursor przyjmie dokładnie taką postać, jaką przedstawiono we wzorcu. Zamiast rozmyślać, lepiej wykonać kilka eksperymentów. Poniższy program może posłużyć za punkt wyjścia. Po uruchomieniu wypełnia on ekran przypadkowymi odcinkami i oczekuje na naciśnięcie któregoś z przycisków myszki. Lewy - nadaje myszce taką postać, jak zaprojektowaliśmy powyżej. Przycisk prawy powoduje zastosowanie maski złożonej z samych jedynek (kursor inwersyjny), natomiast przycisk środkowy uaktywnia maskę z samych zer, dającą kursor w pustym prostokącie. Równoczesne naciśnięcie lewego i prawego przycisku kończy pracę programu:

```
VAR X, Y, Xc, Yc, St, i: Integer;
CONST  Wzorzec: Raster_kursora=
         ($0000,$4000,$7000,$7C00,$7F00,$7FC0,$7E00,$6600,
          $4300,$0180,$00C0,$0060,$0030,$0018,$000C,$0000);
  Maska1: Raster_kursora=
        ($3FFF,$0FFF,$03FF,$00FF,$003F,$000F,$007F,$00FF,
         $187F,$FC3F,$FE1F,$FF0F,$FF87,$FFC3,$FFE1,$FFF9);
  Maska2: Raster_kursora=
      ($FFFF,$FFFF,$FFFF,$FFFF,$FFFF,$FFFF,$FFFF,$FFFF,
       $FFFF,$FFFF,$FFFF,$FFFF,$FFFF,$FFF3,$FFFF,$FFFF);
  Maska3: Raster_kursora=
      (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0);
BEGIN
  Hires;
  IF NOT Mysz_obecna THEN BEGIN Write('Brak myszy');
                                             Halt END;
  FOR i:= 1 TO 100 DO
```

```
    Draw(Random(639),Random(199), Random(639),Random(199), 1);
    Pokaz_kursor; Xc:= 1; Yc:= 1;
    REPEAT Pozycja_kursora(X, Y, St);
            IF (St AND 7)=1 THEN
                Definiuj_kursor_graf(Xc, Yc, Wzorzec, Maska1);
            IF (St AND 7)=2 THEN
                Definiuj_kursor_graf(Xc, Yc, Wzorzec, Maska2);
            IF (ST AND 7)=4 THEN
                Definiuj_kursor_graf(Xc, Yc, Wzorzec, Maska3);
    UNTIL (St AND 3)=3
END.
```

Jedną z najatrakcyjniejszych właściwości handlera myszki jest możliwość programowania asynchronicznej obsługi związanych z myszką zdarzeń. Zdarzenia te wywołują przerwania, w wyniku których wywołana może zostać przygotowana przez użytkownika procedura. Możliwe zdarzenia to: przemieszczenie myszki oraz naciśnięcie lub zwolnienie któregoś z przycisków. Aby uruchomić obsługę zdarzenia, należy przekazać firmowemu programowi obsługi myszki adres naszego programu obsługi napisany w Turbo Pascalu oraz informację, na które z możliwych zdarzeń należy zareagować. Służy do tego funkcja usługowa nr 12. W parze ES:DX przekazujemy adres, natomiast w CX - tzw. maskę zdarzeń. Każdy bit tej maski jest przyporządkowany innemu zdarzeniu. Jego ustawienie oznacza, że wystąpienie danego zdarzenia spowoduje wywołanie wcześniej zdefiniowanej procedury. Znaczenia poszczególnych bitów zebrane zostały w tabeli 2.

Tabela 2.

Numer bitu	Zdarzenie
0	Przemieszczenie kursora
1	Wciśnięcie lewego przycisku myszki
2	Zwolnienie lewego przycisku myszki
3	Wciśnięcie prawego przycisku myszki
4	Zwolnienie prawego przycisku myszki
5	Wciśnięcie prawego przycisku myszki (jeśli jest)
6	Zwolnienie prawego przycisku myszki (jeśli jest)
7..15	Nie używane

Pisząc procedurę obsługi zdarzeń należy mieć na uwadze, że jest ona wywoływana z handlera rozkazem maszynowym CALL FAR, powinna zatem kończyć się rozkazem RET FAR (nie IRET!). Wprawdzie najbardziej odpowiednim językiem do pisania procedur obsługi może wydawać się Asembler, to jednak przy odrobinie dobrych chęci można posłużyć się także np. Turbo Pascalem w wersji 3.0. Trzeba jednak uwzględnić kilka okoliczności. Po pierwsze, procedury w Turbo Pascalu 3.0 są wywoływane rozkazem CALL NEAR (adres jest nie cztero- lecz jedynie dwubajtowy, w dodatku nie jest to adres absolutny, lecz względny), powrót następuje w wyniku rozkazu RET NEAR. Po drugie - to już problem bardziej ogólny - kod maszynowy skompilowanej procedury pascalowej rozpoczyna się tzw. prologiem (rezerwacja na stosie pamięci dla zmiennych lokalnych itd.) i kończy tzw. epilogiem (oczyszczenie stosu itd.). Oprócz tego trzeba zadbać o przechowanie zawartości rejestrów procesora. Napisaną w Turbo Pascalu procedurę obsługi należy więc uzupełnić na początku i na końcu odpowiednimi wstawkami w języku maszynowym, zapisanymi w formie instrukcji INLINE. Oto ogólna struktura części operacyjnej takiej procedury:

```
{$U-}
BEGIN
   INLINE($50/$53/$51/$52/$57/$56/$06);
   ..................... instrukcje .......................
   INLINE($07/$5E/$5F/$5A/$59/$5B/$58/$8B/$E5/$5D/$CB)
END;
```

Dla dociekliwych asemblerowy listing obydwu wstawek:

```
PUSH AX        POP ES
PUSH BX        POP SI
PUSH CX        POP DI
PUSH DX        POP DX
PUSH DI        POP CX
PUSH SI        POP BX
PUSH ES        POP AX
               MOV SP,BP
               POP BP
               RET FAR
```

Procedura obsługi nie może mieć parametrów. **Uwaga!** Procedurę obsługi należy obowiązkowo skompilować przy wyłączonym przełączniku U (dyrektywa {$U-}). Jeśli tego nie zrobimy, kompilator wstawi jeszcze przed kodem instrukcji INLINE wywołanie procedury testowania klawiatury, której wykonanie spowoduje w najlepszym razie efektowną awarię programu, a w najgorszym - trudne do zlokalizowania, przypadkowe błędy, związane z niekontrolowaną modyfikacją zawartości pamięci! Jeżeli pożądana jest aktywność przełącznika U w pozostałej części programu, można bezpośrednio za procedurą obsługi włączyć go ponownie dyrektywą {$U+}.

Uaktywniać procedurę obsługi można inną procedurą pascalową:

```
PROCEDURE Uruchom_obsluge_przerwania(Adres, Maska:Integer);
VAR Rejestry: Rejestry_8088;
BEGIN Rejestry.AX:= 12;       Rejestry.CX:=Maska;
      Rejestry.DX:= Adres;    Rejestry.ES:= Cseg;
      Intr(51, Rejestry)
END;
```

Przy wywołaniu tej procedury jako pierwszego parametru należy użyć wyrażenia postaci: Ofs(Procedura_obslugi), dostarczającego przesunięcia adresowego danej procedury (segment kodu jest wspólny dla wszystkich procedur programu w Turbo Pascalu 3.0).

Przed zakończeniem pracy programu należy obowiązkowo procedurę obsługi przerwania wyłączyć. Po zakończeniu pracy w miejsce procedury obsługi może zostać załadowana inna zawartość, a wtedy .... W celu wyłączenia procedury obsługi wystarczy wywołać funkcję nr 12 z zerową maską. Oto odpowiednie narzędzie w Turbo Pascalu:

```
PROCEDURE Skasuj_obsluge_przerwania;
VAR Rejestry: Rejestry_8088;
BEGIN Rejestry.AX:= 12; Rejestry.CX:= 0;
      Intr(51, Rejestry)
END;
```

Programowanie obsługi przerwań w Turbo Pascalu to materiał na osobny artykuł. Tutaj starczy miejsca tylko na zwięzłe omówienie wymagań i ograniczeń, które procedura musi spełnić. Wynikają one głównie z następującego faktu: jedynym rejestrem, którego zawartość jest możliwa do przewidzenia w chwili wywołania procedury obsługi, jest CS. Rejestr DS wcale nie musi wskazywać na pascalowy segment danych, zaś para: SS:SP na szczyt pascalowego stosu! Łączność z resztą programu pascalowego mogą zapewnić tylko tryby adresacji, odwołujące się do rejestru CS. Tryby te są stosowane tylko do zmiennych o wstępnie określonej wartości (ang. *typed constant*), deklarowanych w instrukcji CONST. Zmienne te są bowiem lokowane w segmencie kodu programu. Co prawda ograniczenia te można wyeliminować, wymagałoby to jednak znacznego skomplikowania maszynowych "wstawek".

Procedurze obsługi nie wolno odwoływać się do żadnych zmiennych deklarowanych po VAR - nawet do globalnych. Wyjątkiem są zmienne lokalne procedury obsługi. W praktyce lepiej jednak unikać w procedurze obsługi deklaracji VAR i zmienne lokalne deklarować także po CONST. Zmniejsza to zapotrzebowanie na stos (w chwili wywołania procedury nie wiemy, jaką mamy rezerwę na stosie). Z tego samego powodu należy unikać w procedurze obsługi wywoływania innych procedur, zdefiniowanych w tym programie.

Ostrożności wymaga nawet odwoływanie się do procedur standardowych. Wadliwie mogą działać procedury Plot i Draw. Np. wywołanie Write prowadzi na ogół do fatalnych skutków. Jeśli trzeba coś wyprowadzić na ekran, lepiej już wpisywać dane wprost do pamięci ekranu lub odwołać się bezpośrednio do procedur usługowych BIOS. Natomiast do procedur: Sound, NoSound i Delay oraz do funkcji i procedur arytmetycznych można odwoływać się bez ryzyka (w każdym razie autor nie miał tu przykrych doświadczeń, teoretyczne rozważania także nie wskazują na zagrożenie). Bez obawy można natomiast korzystać z funkcji usługowych handlera myszki, np. w celu ustalenia jej współrzędnych kursora stanu przycisków.

Oto przykład obsługi zdarzeń w Turbo Pascalu. Program włącza tryb graficzny i wypisuje w pętli ciąg liczb. Uaktywniona wcześniej procedura obsługi jest wywoływana po każdym przemieszczeniu myszki (ustawiony bit nr 0 w masce). Jeśli był przy tym wciśnięty lewy przycisk, myszka pozostawia na ekranie ślad (funkcja $0C przerwania BIOS 10H powoduje wypełnienie punktu o współrzędnych X, Y w rejestrach CX i DX, nadając mu barwę o kodzie w AL). Każde przemieszczenie myszki jest kwitowane krótkimi dźwięka-

> 34

33 <

mi, których wysokość zależy od tego, czy lewy przycisk jest wciśnięty. Dlaczego współrzędna Y w rejestrze DX jest zmniejszana o 2? Otóż chodzi tu o narysowanie punktu poza rastrem kursora myszki.

Wszystkie zmienne robocze procedury obsługi zadeklarowano jako obiekty o wstępnie określonej wartości:

```
CONST Wsk_konca: Boolean= FALSE;
{.......... Inne definicje i deklaracje .........}
{$U-}
PROCEDURE Obsluga_przerwania;
CONST F : Integer=0;
    Rejestry: Rejestry_8088=(AX:0; BX:0; CX:0; DX:0;
                              BP:0; SI:0; DI:0; DS:0; ES:0);
BEGIN
  INLINE($50/$53/$51/$52/$57/$56/$06);
  WITH Rejestry DO
    BEGIN AX:= 3; Intr(51, Rejestry);
      IF (BX AND 1)>0
        THEN BEGIN DX:= DX-2; AX:= $0C01; F:= 4400;
          Intr($10, Rejestry);
          END
        ELSE F:= 1200;
      IF (BX AND 2)>0 THEN Wsk_konca:= TRUE;
    END;
  Sound(F); Delay(1); NoSound;
  INLINE($07/$5E/$5F/$5A/$59/$5B/$58/$8B/$E5/$5D/$CB)
END; {$U+}
{Tu definicje proc.: Uruchom i Skasuj_obsluge_przerwania}
VAR i: Integer;
    X: Real;
BEGIN Wsk_konca:= FALSE;
  IF NOT Mysz_obecna THEN Exit;
  Hires; Pokaz_kursor; X:= 1;
  Uruchom_obsluge_przerwania(Ofs(Obsluga_przerwania),9);
  REPEAT
    FOR i:= 1 TO 25 DO
      BEGIN GotoXY(1,i); Write(X:7:0); X:= X+1 END;
  UNTIL Wsk_konca;
  Skasuj_obsluge_przerwania; TextMode;
END.
```

Równolegle z pracą programu zasadniczego można sporządzać na ekranie proste szkice. Program kończy pracę po naciśnięciu prawego przycisku myszki. Zachodzi tu przekazywanie danej od procedury obsługi do programu głównego, co odbywa się za pośrednictwem zlokalizowanej w segmencie kodu zmiennej globalnej: Wsk_konca.

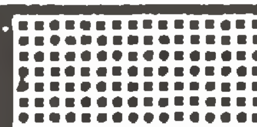
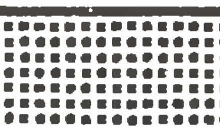
Prowadząc własne eksperymenty z obsługą przerwań trzeba postępować ostrożnie i zapisywać program przed każdym uruchomieniem. Cierpliwość może się jednak opłacić.

Na zakończenie warto wspomnieć, że myszka może zastąpić (emulować) pióro świetlne. Do pióra świetlnego odwołujemy się za pośrednictwem funkcji usługowej nr 4 przerwania nr 16. Funkcja nr 13 przerwania 51 włącza, natomiast funkcja nr 14 - wyłącza tryb emulacji pióra świetlnego. Przy wywołaniu funkcji nr 13 nie są przekazywane żadne parametry, natomiast funkcja nr 14 wymaga podania w rejestrach CX i DX nowych współrzędnych kursora. Do włączania i wyłączania emulacji pióra świetlnego w Turbo Pascalu można użyć poniższej procedury (TRUE=włączenie, FALSE=wyłączenie emulacji):

```
PROCEDURE Wlaczenie_emulacji_piora;
VAR Rejestry: Rejestry_8088;
BEGIN Rejestry.AX:= 13; Intr(51, Rejestry) END;

PWOCEDURE Wylaczenie_emulacji_piora(WspX, WspY: Integer);
VAR Rejestry: Rejestry_8088;
BEGIN WITH Rejestry DO
          BEGIN AX:= 14; CX:= WspX; DX:= WspY END;
    Intr(51, Rejestry)
END;
```

W praktyce emulacja pióra świetlnego jest raczej ciekawostką, gdyż większość programów pozwalających wykorzystać pióro świetlne może też współpracować z myszką, dając w tym drugim przypadku większy komfort obsługi.



W pracy



Krzysztof Kontek




Abel – język programowania układów PLD






Skrót PLD (ang. *Programmable Logic Devices* - układy o programowalnej logice) oznacza układy o funkcjach programowanych przez użytkownika. W klasycznej technologii bipolarnej programuje się je przepalając cienkie połączenia (ang. *fuse* - bezpieczniki) w matrycach bramek w ich wnętrzu. W nowszych technologiach (np. EECMOS) programowanie nie jest niszczące i może być powtarzane.

Do układów PLD należą układy PROM *(Programmable Read Only Memory)*, FPLA *(Field Programmable Logic Array)* oraz PAL *(Programmable Array Logic)*. Zawierają one dwie matryce bramek. Pierwsza (bramki AND) przetwarza sygnały wejściowe na pośrednie, z których druga (bramki OR) tworzy sygnały wyjściowe. Typy układów PLD różni sposób programowania: w układach PROM programuje się matryce OR, w układach PAL - matryce AND, natomiast w FPLA - obie matryce.

W praktyce najczęściej spotykamy układy PROM jako zwykłe pamięci stałe. Realizują one funkcje wysoce nieregularne, w których istotna jest większość kombinacji sygnałów wejściowych. Gdy sygnałów wejściowych jest dużo, a realizowana funkcja dla wielu ich kombinacji nieokreślona (co zdarza się często), użycie PROM staje się ekonomicznie nieuzasadnione: do uzyskania ośmiu funkcji wyjściowych na podstawie 12 sygnałów wejściowych potrzebna jest pamięć 4Kx8 bitów. Pamięci takie są drogie, wolne (typowy czas dostępu 35 - 50 ns) oraz pobierają spory prąd.

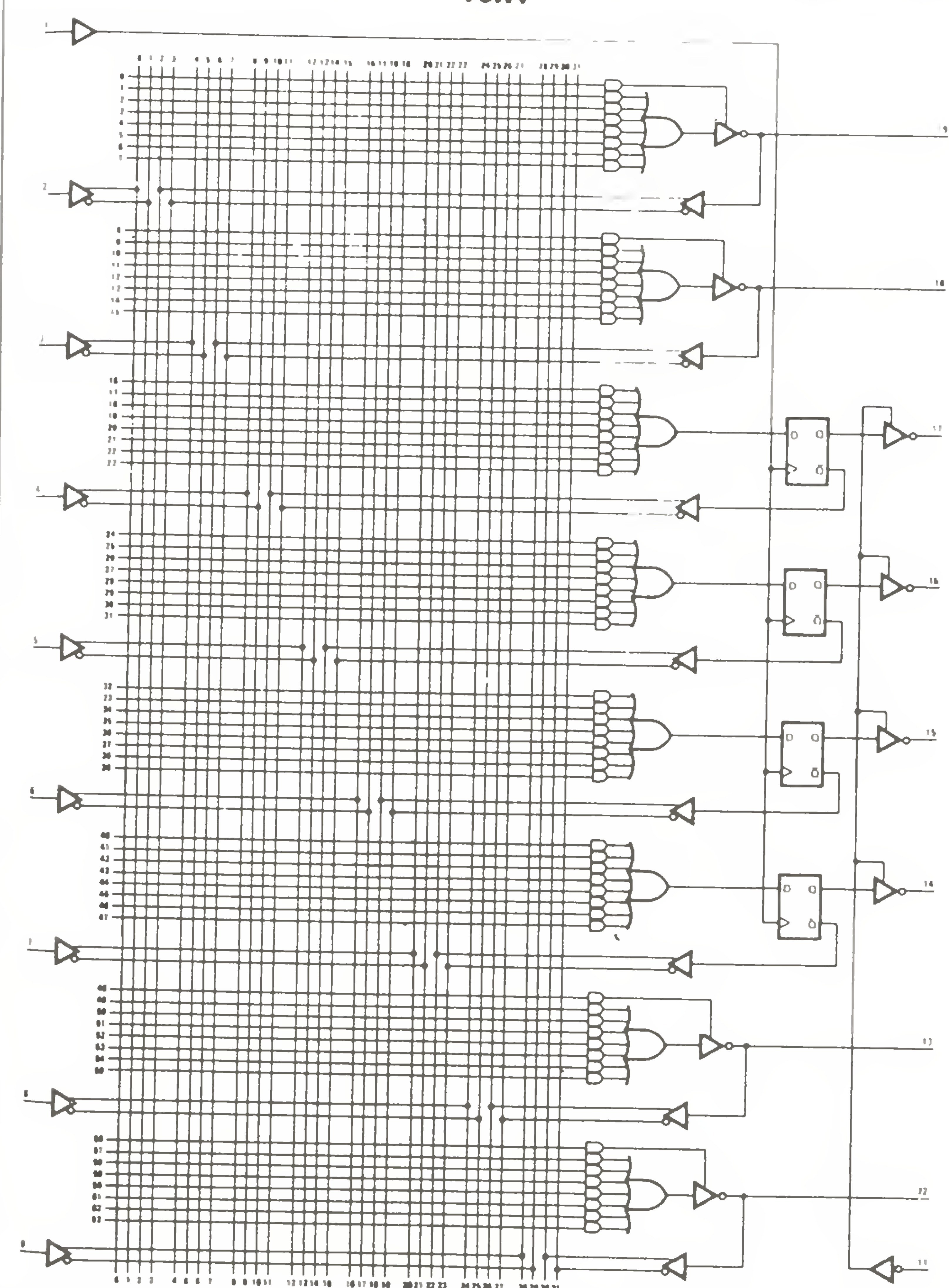
Układy PAL i FPLA realizują funkcje wielu zmiennych na ogół taniej niż pamięci PROM. Układy PAL są szybkie (czas przejścia bywa krótszy od 10 ns), lecz do budowy funkcji wyjściowej można użyć zwykle tylko ośmiu iloczynów i każdy z nich może służyć realizacji tylko jednej funkcji. Ograniczeń tych nie mają układy FPLA: do budowy funkcji można użyć dowolnej liczby iloczynów, a każdy z nich - w wielu funkcjach wyjściowych. Są one jednak powolne i wymagają drogich specjalizowanych programatorów. Rynek zdobywają więc układy PAL. Ich ograniczenia można ominąć i programować je za pomocą typowych programatorów pamięci PROM.

Układ GAL39V18

Pod koniec 1987 r. pojawił się układ GAL39V18 firmy Lattice, łączący elastyczność FPLA z szybkością i łatwością programowania układów PAL. W 24-nóżkowej obudowie zawiera on 64 bramki AND, z których każda może tworzyć funkcje 39 zmiennych. Można je dowolnie sumować dla uzyskania 18 sygnałów wyjściowych. Sygnały wejściowe i wyjściowe oraz niektóre pośrednie przechodzą przez 38 tzw. makrokomórek (ang. *macrocell*). Każda z nich może być skonfigurowana jako bramka, przerzutnik typu **latch** lub **D**, a 18 z nich także jako przerzutniki **J-K**, T lub **SR**. Makrokomórki mogą korzystać ze wspólnego lub własnego zegara. Wszystkie sygnały wejściowe i wyjściowe makrokomórek i matrycy AND mają wybieralną polaryzację, a sygnały wyjściowe mogą być z powrotem doprowadzone do bramek AND. Układ jest bardzo szybki (typowy czas przejścia przez kilka poziomów bramek 20 ns), pobiera małą moc (400 mW) i może być przeprogramowany w czasie krótszym od 1s za pomocą typowego programatora układów PAL.

Projektowanie i programowanie układów PLD

Pierwsze układy PROM pojawiły się ok. 1970 r., w kilka lat później - układy FPLA, a ok. 1978 r. - układy PAL. Choć każdy z nich zastąpić może od kilku do kilkunastu układów TTL małej i średniej skali integracji, nowa technologia początkowo nie zyskała popula-



Architektura wnętrza typowego układu PLD

rności. Przyczyną był m.in. brak programów wspomagających ich projektowanie. Projektant musiał więc sam określić minimalną postać realizowanej funkcji (na przykład z tablic Karnaugh) i mozolnie wyznaczać listę kilku tysięcy przepaleń w układzie - jak w czasach, gdy programiści dziurkowali karty z poleceniami dla komputera.

Pierwsze programy wspomagające projektowanie układów PLD - np. PALASM (PAL ASeMbler) firmy MMI, pierwszego producenta układów PAL, napisany w Fortranie dla komputerów serii VAX - mimo skromnych możliwości pozwoliły inżynierom - tak jak niegdyś asemblery - oderwać się od dziurek i zająć ogólniejszymi problemami. Projektant nie musiał już tworzyć tablic przepaleń i zbioru danych dla programatora, ale nadal musiał stosować tablice Karnaugh do optymalizacji. PALASM pozwalał programować tylko układy PAL firmy MMI (do pamięci PROM służył program PLEASM). Inne firmy również oferowały osobne programy dla własnych układów PAL, FPLA i PROM.

Stan rynku PLD ok. 1982 r. przypominał stan informatyki przed powstaniem sprawnych kompilatorów języków wysokiego poziomu, gdy stosowanie asemblerów utrudniało przenoszenie oprogramowania (problem znany osobom pragnącym przenieść np. gry napisane w asemblerze Z 80 dla ZX Spectrum do Atari XL lub choćby do opartego również na Z 80 Amstrada CPC). Języki takie jak Fortran, Cobol lub Pascal pozwoliły rozwiązywać problemy niezależnie od stosowanego komputera. Tak też się stało na rynku PLD. W latach 1983-1987 pojawiły się kompilatory języków wysokiego poziomu do opisu układów cyfrowych oraz projektowania i programowania układów PLD.

Język ABEL - budowa

Do najpopularniejszych tego typu programów należą ABEL firmy Data I/O oraz CUPL firmy Assisted Technology. Są one wzorowane na języku C i mają podobną składnię i możliwości. Tu dokładniej opiszemy język ABEL.

Tekst źródłowy programu w języku ABEL można przygotować za pomocą dowolnego edytora tekstu, np. Wordstara, jednak firma DATA I/O zaleca program PC-Write. Tekst ten jest kompilowany sześcioma osobnymi programami.

Program PARSE sprawdza poprawność składni wyrażeń, definicji i makrodefinicji, wprowadza makrodefinicje do tekstu źródłowego oraz przekształca postać opisu realizowanych funkcji z tablic prawdy i diagramów na równania logiczne.

Program TRANSFOR zastępuje w równaniach logicznych zdefiniowane zbiory odpowiednimi równaniami oraz wszystkie operatory operatorami NOT, AND, OR i XOR, a także wstępnie redukuje równania według zasad logiki boolowskiej.

Program REDUCE dalej redukuje równania według specjalnego algorytmu opracowanego przez Antonina Svobodę, umożliwiając wybór redukcji optymalnej dla realizacji za pomocą FPLA lub PAL (w przypadku PROM redukcja nie jest konieczna).

Program FUSEMAP tworzy tablice przepaleń dla wskazanego urządzenia PLD oraz zbiór wynikowy dla programatora, do którego dołącza zadeklarowane w programie źródłowym wektory testowe dla programu SIMULATE.

Program DOCUMENT sporządza dokumentację, obejmującą otrzymane równania, tabelę przepaleń, rysunek układu scalonego z opisem sygnałów na poszczególnych wyprowadzeniach, przebieg symulacji oraz opis błędów, które wystąpiły podczas kompilacji. Jeżeli kompilacja przebiegła prawidłowo, projektantowi pozostaje przesłanie łączem szeregowym zbioru wynikowego do programatora i zaprogramowanie układu scalonego.

Zalety i wady języka

Dla projektanta systemów cyfrowych języki ABEL i CUPL mają wiele zalet:

- możliwość opisu układu w sposób wygodny dla projektanta i zgodny z jego przyzwyczajeniami, a więc za pomocą równań logicznych, tablic prawdy, diagramów lub schematów ideowych (przy wykorzystaniu programu do rysowania schematów DASH);
- opis realizowanego układu na wysokim poziomie abstrakcji w sposób jednakowy dla wszystkich układów PLD, co ułatwia zmianę sposobu realizacji projektu np. z konkretnego układu PROM na układ FPLA lub z jednego na inny układ PAL;
- możliwość opisu całego systemu cyfrowego złożonego z wielu różnych układów PLD, co pozwala na szybkie i wygodne przemieszczanie funkcji między zastosowanymi układami;
- możliwość symulacji zaprojektowanego układu;
- dostępność dla komputerów zgodnych z IBM XT/AT oraz komputerów serii VAX działających pod nadzorem systemu operacyjnego VMS lub UNIX;
- duża szybkość działania (pełna kompilacja łącznie z symulacją trwa zwykle krócej niż 1 min.).

Efektem tych udogodnień jest gwałtowne skrócenie czasu upływającego od pomysłu do gotowego, sprawnego układu z pełną dokumentacją - na ogół nie przekracza on kilku godzin, a pewna wprawa pozwala skrócić go do kilkudziesięciu minut. Oznacza to wielokrotne przyspieszenie w porównaniu z projektowaniem urządzeń z układów TTL. Dochodzi do tego wielokrotne zmniejszenie liczby użytych układów scalonych, co zwiększa niezawodność systemu i zmniejsza pobór mocy oraz ułatwia zmianę funkcji układu w razie zmiany założeń.

Język ABEL umożliwia opis układu m.in. za pomocą schematu ideowego. Kompilator przekształca schemat w równania logiczne i projektuje układ PLD. Jest to przydatne, gdy schemat urządzenia jest gotowy i chcemy zastąpić grupę rozrzuconych bramek logicznych układami PLD. Rozpoczynanie projektu od schematu odwraca jednak uwagę od problemów ogólnych ku wyborowi bramek, dobre jest więc dla projektujących układy cyfrowe "na pająka", sięgających bez analizy założeń po układy TTL i lutownicę.

ABEL ma też pewne wady. Mnie kłopoty sprawiają:

1. Ograniczenia w tworzeniu złożonych wyrażeń strukturalnych przy opisie układu za pomocą diagramów.
2. Brak możliwości opisu układu za pomocą diagramów do projektowania asynchronicznych układów sekwencyjnych.
3. Brak możliwości eliminowania ryzyka wystąpienia hazardu (stanów nieustalonych) przy automatycznej redukcji równań logicznych oraz przedstawienia zredukowanych równań w postaci tablicy Karnaugh, co umożliwiłoby "ręczne" jego wyeliminowanie (uwagi 2 i 3 nie dotyczą pamięci PROM, które nie mogą zapewnić bezhazardowych rozwiązań i nie można ich używać do budowy automatów asynchronicznych).

Program SIMULATE

Nieco więcej uwagi warto poświęcić będącemu częścią kompilatora języka ABEL programowi SIMULATE. W istocie nie jest to program symulujący, lecz testujący działanie układu. Projektant podając wektory testowe, czyli możliwe stany wejść układu, musi podać spodziewane stany jego wyjść. Program SIMULATE porów-

> 36

35 <

nuje wyniki symulacji z podanymi założeniami, sygnalizując niezgodności wynikające z błędnego opisu funkcji układu lub błędnego przewidywania wyników symulacji. Zmusza to do nieco większego wysiłku i z początku jest irytujące, w zamian jednak:

- Umożliwia szybkie sprawdzenie, czy układ realizuje założone funkcje po niewielkiej zmianie jego opisu (klasyczny symulator zmusza tu do samodzielnej analizy dużej liczby danych, co prowadzi czasem do przeoczenia błędnego stanu jednego z wyjść np. po usunięciu błędu występującego na innym wyjściu dla innych danych);
- Przesyła wektory testowe do programatora, co pozwala sprawdzić układ po zaprogramowaniu lub przy podejrzeniu o niesprawność (przetestowanie np. układu sekwencyjnego o kilkudziesięciu stanach dla różnych sekwencji sygnałów wejściowych jest w rzeczywistym systemie cyfrowym niemożliwe lub trudne i czasochłonne). Test taki wymaga odpowiednio przystosowanego programatora - samo sprawdzenie przepaleń nie pozwala stwierdzić, czy układ jest dobrze zaprogramowany! Odczyt przepaleń nie jest zresztą możliwy po zniszczeniu specjalnego bezpiecznika wewnątrz układu.
- Zmusza do precyzyjnego przewidywania wszystkich sytuacji, w jakich może się znaleźć projektowany układ, co pozwala wiele błędów dostrzec samodzielnie.

Wielką zaletą programu SIMULATE jest możliwość analizy układów ze sprzężeniem zwrotnym, w których stany wyjściowe są przejściowo niestabilne oraz możliwość przedstawienia wyników symulacji w formie przebiegów czasowych (jak na ekranie wielokanałowego analizatora stanów logicznych), wadą zaś brak możliwości jednoczesnej symulacji pracy kilku układów PLD opisanych w jednym module oraz niesygnalizowanie ryzyka hazardu na wyjściach układu.

Przykłady

Oto trzy projekty układów ukazujące wybrane możliwości języka ABEL (pierwsze dwa z niewielkimi modyfikacjami zaczerpnięto z "ABEL Applications Guide"):

Przykład 1: dekoder

Załóżmy, że potrzebny jest dekoder adresów dla systemu z mikroprocesorem 6809. Przestrzeń adresowa mikroprocesora ma być podzielona na pięć części: od h0000 do hDFFF (pamięć DRAM), od hE000 do hE7FF (urządzenia I/O), od hE800 do hEFFF (nie używana), od hF000 do hF7FF (pamięć ROM2) oraz od hF800 do hFFFF (pamięć ROM1).

Program opisujący układ w języku ABEL rozpoczyna nazwa modułu, krótki opis, nazwa układu użytego do realizacji dekodera (w tym wypadku PAL P14L4) oraz proponowany rozkład wyprowadzeń - projektant odpowiednio wyprowadzając sygnały może ułatwić sobie prowadzenie ścieżek na płytce. Kompilator sprawdza, czy rozkład nie ma błędów, np. czy nie pomylono wejść i wyjść. Dalej w programie zdefiniowano słowo Address. Może być ono rozumiane jako 16-elementowy zbiór (ABEL pozwala na wykonywanie operacji na zbiorach) lub też jako liczba o wartości określonej przez elementy zbioru (tu przez stan wierszy adresowych). Po słowie kluczowym "equations" podano równania opisujące dekoder, przy czym ! oznacza negację, => - większy lub równy, # - sumę logiczną, a & - iloczyn logiczny. Przy określaniu wyprowadzeń zadeklarowano sygnały zanegowane, więc stanem aktywnym (wskazującym na wybór danego obszaru pamięci) jest poziom niski.

W otrzymanej po kompilacji dokumentacji podana jest ostateczna postać równań. Projektant mógłby podać ją w tekście źródłowym, byłby on wtedy znacznie mniej czytelny.

```
PRZYKŁAD 1
module m6809a
title '6809 memory decode
      Jean Designer Data I/O Corp Redmond WA 24 Feb 1984'
"device declaration
      IC1    device  'P14L4';
"pin declaration
      A15,A14,A13,A12,A11,A10 pin 1,2,3,4,5,6;
      !ROM1,!ROM2,!IO,!DRAM   pin 14,15,16,17;
"constant declaration
      H,L,X   = 1,0,.X.;
      Address = [A15,A14,A13,A12, A11,X,X,X, X,X,X,X, X,X,X,X];
equations
      DRAM   = (Address <= ^hDFFF);
      IO     = (Address >= ^hE000) & (Address <= ^hE7FF);
      ROM2   = (Address >= ^hF000) & (Address <= ^hF7FF);
      ROM1   = (Address >= ^hF800);
end m6809a
```

Przykład 2: transkoder

Pokazuje on opis układu za pomocą tablicy prawdy. Transkoder wskaźnika siedmiosegmentowego przypomina układ TTL 7449, lecz jego funkcje można łatwo rozszerzyć np. o wyświetlanie liter A, B, C, D, E, F, gdy liczba wejściowa jest większa od 9. Zrealizowano go za pomocą pamięci PROM RA5P8 (5 wejść, 8 wyjść, pojemność 32x8 bitów). Sygnały wyjściowe zapalają segmenty wskaźnika, definiujemy więc stałe ON i OFF równe 0 i 1 (poziom niski zapala segment).

Po słowie kluczowym "truth_table" podano tabelę stanów segmentów dla poszczególnych cyfr. Opis ich za pomocą równań logicznych byłby trudny.

Transkoder ten można zrealizować także za pomocą innego układu, np. PAL P16L8 - wystarczyłoby zmienić deklarację we wstępnej części programu. Opis funkcji transkodera w języku ABEL pozostałby niezmieniony.

```
PRZYKŁAD 2.
module bcd7rom flag '-d82'
title 'seven segment display decoder
      Data I/O Corp   Redmond WA   16 Mar 1984'
"  ---         ---   ---         ---   ---   ---    -a-    ---
" |   |     |     |     | |   | |     |        | f   b |     |
"               ---   ---   ---   ---   ---          -g-    ---
" |   |     |  |           |     |     | |   |     | e   c       |
"  ---         ---   ---         ---   ---          -d-    ---
"device declaration
     IC2  device  'RA5P8';
"pin declaration
"    inputs
     D3,D2,D1,D0       pin 10,11,12,13;
"    outputs
     a,b,c,d,e,f,g     pin 1,2,3,4,5,6,7;
     ena               pin 15;
"set declaration
     bcd   = [D3,D2,D1,D0];
     led   = [a,b,c,d,e,f,g];
"constant declaration
     ON,OFF  = 0,1;    " for common anode LEDs
     L,H,X,Z = 0,1,.X.,.Z.;
truth_table (bcd  -> led)
"  input                outputs
"            a    b    c    d    e    f    g
     0 -> [ ON,  ON,  ON,  ON,  ON,  ON, OFF];
     1 -> [OFF,  ON,  ON, OFF, OFF, OFF, OFF];
     2 -> [ ON,  ON, OFF,  ON,  ON, OFF,  ON];
     3 -> [ ON,  ON,  ON,  ON, OFF, OFF,  ON];
     4 -> [OFF,  ON,  ON, OFF, OFF,  ON,  ON];
     5 -> [ ON, OFF,  ON,  ON, OFF,  ON,  ON];
     6 -> [ ON, OFF,  ON,  ON,  ON,  ON,  ON];
     7 -> [ ON,  ON,  ON, OFF, OFF, OFF, OFF];
     8 -> [ ON,  ON,  ON,  ON,  ON,  ON,  ON];
     9 -> [ ON,  ON,  ON,  ON, OFF,  ON,  ON];
end bcd7rom
```

Przykład 3: sterownik

Język ABEL pozwala projektować również układy sekwencyjne, w których stan wyjść zależy nie tylko od sygnałów wejściowych, lecz także od stanu danego układu. Przykładem jest fragment sterownika kasety FASTBUS, opracowanego w ramach eksperymentu LEAR w CERN-ie w Genewie. Do jego zadań należy m.in. przesłanie danych z kasety do komputera sterującego eksperymentem.

We wstępnej części programu określono stany sterownika: Begin, Transfer, No_Data, Error, Stop. Przejścia między stanami określono po słowie kluczowym **"state_diagram"**. Dla znających Pascal lub C opis jest czytelny: gdy blok zbiera dane (BUSY), pozostań w stanie Begin; jeżeli nie jest zajęty, ale wystąpił błąd (ERR), przejdź do stanu Error; jeżeli błąd nie wystąpił i blok zawiera dane do odczytania (DATA), przejdź do stanu Transfer, gdy zaś danych nie ma, przejdź do stanu No_Data itd. Ten prosty opis wystarcza do zaprojektowania sterownika realizującego podany algorytm.

Czy PLD zastąpią mikroprocesory?

Z pozoru układów PLD nic nie wiąże z mikroprocesorami. Podobieństw jest jednak wiele. W obu przypadkach układy scalone realizują sprzętowo algorytm określony w zbliżonych językach programowania. Różni je to, że mikroprocesory interpretują program szeregowo, a układy PLD - równolegle.

Budowę wewnętrzną mikroprocesora określa producent. Składają się nań uniwersalne bloki, takie jak rejestry i układy arytmetyczne, łączone przez wewnętrzne multipleksery zgodnie z instruk-

```
PRZYKŁAD 3.
module controller
title 'FASTBUS Controller
       Krzysztof Kontek, Experimental Electronic Group,
       Schweizerisches Institut fuer Nuklearforschung, Villigen,
       (on leave from IPJ,Swierk,Poland) 30.09.87.'
"device declaration
       IC3  device   'P16R4';
"pin declaration
"      inputs in IC3
       !NEXT_BLOCK,!ERR,!DATA      pin   2,3,4;
       !MESSAGE_ACK,!BUSY          pin   5,6;
"      outputs in IC3
       Q2,Q1,Q0                    pin   17,16,15;
       !NO_DATA_MESSAGE            pin   19;
       !ERROR_MESSAGE              pin   18;
       !TRANS                      pin   12;
"constant declaration
       Controller_State = [Q2,Q1,Q0];
       Begin            = [ 1, 1, 1];
       Transfer         = [ 1, 0, 0];
       No_Data          = [ 0, 1, 0];
       Error            = [ 0, 0, 1];
       Stop             = [ 0, 1, 1];
"state diagram
 state_diagram  Controller_State
state Begin:    case   BUSY                      :Begin;
                      !BUSY &  ERR               :Error;
                      !BUSY & !ERR &  DATA :Transfer;
                      !BUSY & !ERR & !DATA :No_Data;
                 endcase;
state Transfer: TRANS = 1;
                case  DATA & !ERR             :Transfer;
                     !DATA & !ERR             :No_Data;
                      ERR                     :Error;
                endcase;
state No_data:  NO_DATA_MESSAGE = 1;
                if MESSAGE_ACK then Stop
                               else No_data;
state Error:    ERROR_MESSAGE = 1;
                if MESSAGE_ACK then Stop
                               else Error;
state Stop:     if NEXT_BLOCK then Begin
                              else Stop;
end controller
```

cjami programu. Pobranie instrukcji, zinterpretowanie jej oraz zmiana połączeń trwa, opóźniając realizację programu. Pamięci podręczne (ang. *cache*) dla instrukcji, wstępne pobieranie i interpretację instrukcji (ang. *prefetching*) lub rozdzielenie pamięci instrukcji i danych (architektura Harvard) redukują, dzięki wielkim wysiłkom projektantów, tracony czas. Zajmujące się tym wyrafinowane układy stanowią sporą część mikroprocesora, ale problem pozostaje. Uniwersalne bloki są ponadto zawsze wolniejsze od układów specjalizowanych.

Tymczasem układy PLD przed zaprogramowaniem mają luźną strukturę, w której bramki, przerzutniki, a z czasem zapewne także bardziej złożone bloki, łączone są dopiero po zaprogramowaniu. W gotowym do realizacji programu układzie wewnętrzne połączenia między elementami nie zmieniają się. Program jest wykonywany przez sprzęt specjalizowany, a więc szybciej niż przez mikroprocesor. Okupione to jest większą liczbą elementów używanych do bardziej złożonych algorytmów, długość i złożoność programu realizowalnego techniką PLD na danym etapie rozwoju technologii jest więc ograniczona.

Nowości

Technologia jednak rozwija się: 64 makrokomórki układu XC3020 firmy Xilinx z 1988 r. realizują dowolną funkcję pięciu, a czasem i 7 zmiennych, bądź dwie funkcje 4 zmiennych. Zawierają też dwa przerzutniki rozbudowane bardziej niż w GAL39V18. Konfigurację makrokomórek oraz ich połączenia zapisano w automatycznie ładowanej podczas włączania zasilania pamięci RAM. Przeprogramować może ją np. mikroprocesor. Układy te w prostych sterownikach zastępują mikroprocesory i trend ten pogłębia się.

Rozwija się też oprogramowanie. Miałem już okazję pracować z lepszymi niż ABEL systemami do projektowania układów PLD na komputerach IBM PC/AT i VAX. Pozwoliły one zaprojektować specjalizowany sterownik o 70 wejściach i 80 wyjściach zupełnie niezależnie od rodzaju użytego PLD. Układ został następnie przetestowany przez napisany także w języku wysokiego poziomu program testujący. Wygenerował on tysiące wektorów testujących. Projekt następnie podzielono na pięć 40-nóżkowych układów PLD z rozbudowaną architekturą wewnętrzną. Każdy z nich w wersji SMD *(Surface Mounted Devices)* zajmuje ok. 1 cm kw.

W innym projekcie automatycznie przeniesiono schemat z ekranu monitora do wnętrza innego układu PLD. Słychać o systemach do programowania PLD za pomocą zwykłych przebiegów czasowych. Pytanie "jak zaprojektować układ" zastępowane jest pytaniem "jak go opisać".

Rozwój układów PLD stymulowany jest przez sprzężenie zwrotne: języki wysokiego poziomu do ich projektowania zwiększyły zainteresowanie użytkowników, co zachęca producentów kości do tworzenia coraz bardziej rozbudowanych układów, te zaś zmuszają do doskonalenia oprogramowania.

Przewiduje się, że sprzedaż układów PLD (bez klasycznych pamięci PROM) wzrośnie z 235 mln dolarów w 1985 r. i 417 mln w 1987 r. do ponad miliarda dolarów w 1992 r.

Nie ma armat

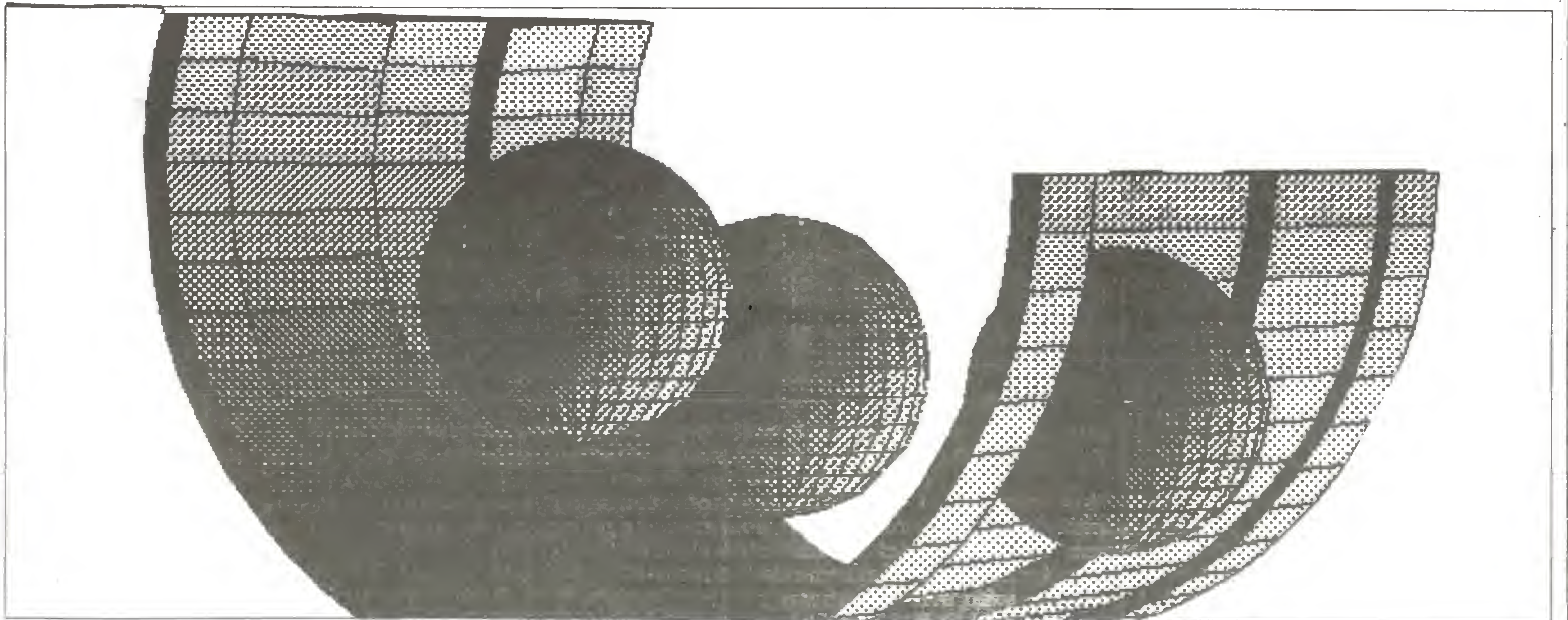
By uwolnić redakcję od pytań: "gdzie jest autor?" w znaczeniu "chciałbym mieć ten program" dodam, że ABEL jest chroniony programowo i sprzętowo przed kopiowaniem. W Polsce trudno go zresztą użyć z braku programatorów oraz... samych układów PLD. Układy PAL i FPLA - o ile wiem - nie są produkowane w RWPG. Miejmy nadzieję, że stan ten się zmieni.

Polecam:

1. PROGRAMMABLE LOGIC DEVICES: FASTER, DENSER, AND A LOT MORE OF THEM Electronics, Sept.17,1987. str. 61-72

2. PAL/PLE PROGRAMMABLE LOGIC HANDBOOK, Monolithic Memories Inc.,1985

3. GAL HANDBOOK, Lattice, 1986.





Zenon Rudak

Olivetti PC 1

Firma Olivetti znana jest na rynku z produkcji całej gamy komputerów, w tym także zgodnych ze standardem IBM PC. Komputery te wyróżniają się bardzo dobrą jakością, elegancką sylwetką i bardzo solidnym wykonaniem. Są niestety dość drogie. Napór dalekowschodnich, masowo produkowanych kopii standardu PC/AT spowodował zmniejszenie zainteresowania wcześniejszą wersją PC/XT. Kopia komputera IBM PC/ /XT staniała na tyle, aby stać się zagrożeniem dla komputerów domowych. Upatrując w tym zjawisku szanse zysku wielu producentów komputerów rozpoczęło produkcję uproszczonych i tanich wersji komputerów tego standardu. Do grona takich producentów dołączyła firma Olivetti. Tak powstał komputer Olivetti PC 1.

Korzystając z informacji marketingowych i popularności produktów innych firm ustalono podstawową konfigurację oraz możliwości mutacji wersji podstawowej. Za wzór ustalania konfiguracji PC 1 posłużył sprzedawany nadal masowo na rynku angielskim komputer Amstrad PC 1512.

Olivetti PC 1 przeznaczony jest dla początkujących użytkowników komputerów profesjonalnych lub dla osób zamieniających posiadany komputer domowy na inny, o większych możliwościach. Takie przeznaczenie spowodowało zainstalowanie w PC 1 kolorowej karty graficznej (CGA). Karta tego typu umożliwia korzystanie z programów - gier.

Komputer PC 1 składa się z dwóch zasadniczych części. Jedną stanowi monitor z podstawą, a drugą - płaska obudowa mieszcząca wszystkie elementy elektroniczne, klawiaturę i napędy dyskowe. Układ elektroniczny komputera umieszczony jest na jednej płycie drukowanej. Zastosowano procesor typu V40 zgodny programowo i sprzętowo z procesorem Intel 8088. Częstotliwość zegara wynosi 4,77 lub 8 MHz. Szybkość pracy zegara przełączana jest w dowolnej chwili za pomocą kombinacji naciśnięcia kilku klawiszy jednocześnie. Szybkość pracy maszyny z zegarem 8 MHz jest o ok. 170 % większa niż oryginału IBM.

Procesor współpracuje z pamięcią operacyjną RAM o pojemności 512 KB. Obszar ten w zasadzie wystarcza do uruchomienia i pracy wielu popularnych programów, ale może być niewystarczający z chwilą stosowania kilku nakładek na system operacyjny. Pamięć ROM z BIOS komputera stanowi układ EPROM o pojemności 8 KB.

Komputer wyposażony jest w klawiaturę różniącą się nieznacznie od klawiatur stosowanych w kopiach standardu PC. Klawisze funkcyjne nie stanowią oddzielnego pola po lewej stronie maszyny, lecz są ułożone w jednym rzędzie na szczycie klawiatury. Ułożenie i wielkość innych klawiszy funkcyjnych (np. Esc, Alt, Ctrl itp.) także nie odpowiada typowej klawiaturze komputera XT. Działanie, czytelność i jakość wykonania klawiatury oceniana jest bardzo wysoko, co stanowi zresztą atut wszystkich komputerów firmy Olivetti. Klawiatura jest częścią obudowy całego komputera, podobnie jak ma to miejsce w komputerach domowych.

Nad klawiaturą umieszczone są napędy dyskowe. W wersji najtańszej zainstalowany jest tylko jeden napęd dyskietek. W PC 1 zainstalowano napędy dyskietek 3,5-calowych. Dyskietki mogą być zapisywane do pojemności 720 KB. Zastosowanie dyskietek 3,5-calowych ma umożliwić wymianę informacji z komputerami IBM PS/2 oraz bardzo liczną grupą komputerów przenośnych standardu PC.

Sterownik kolorowej karty graficznej wmontowany w płytę główną współpracuje z monitorem kolorowym RGB. W wersji podstawowej karta steruje monitorem monochromatycznym. Konstruktorzy zastosowali monitory o przekątnej ekranu 12 cali. Czytelność monitorów jest dobra dla pracy z programami graficznymi. Przy pracy z tekstem uwidaczniają się wszystkie wady zastosowanego sterownika obrazu. Wyraźniejszy obraz tekstowy uzyskiwany jest na monitorze kolorowym. Aby nie podnosić ceny urządzenia, nie przewidziano stosowania przełączalnego wielofunkcyjnego sterownika obrazu (CGA, Hercules, EGA), jaki zastosowano np. w komputerze Amstrad PC 1640 lub Atari PC.

Komputer PC 1 wyposażono w interfejs równoległy typu Centronics dla drukarki, szeregowy typu RS 232 dla podłączenia modemu, plotera lub digitizera. Zastosowano oddzielne łącze z szyną systemu dla myszki i joysticka. Standardowo komputer wyposażony jest w myszkę oraz w jedno łącze dla kart rozszerzenia standardu IBM PC/XT. Ze względu na małe gabaryty obudowy komputera dodatkowa karta rozszerzenia nie może być schowana wewnątrz maszyny. Producent przewidział możliwość uzupełniania konfiguracji PC 1. Oferowane są przystawki pozwalające na instalowanie dodatkowych kart, łączone przewodem ze złączem systemowym komputera. Przewidziano także możliwość współpracy PC 1 ze stacją dyskietek 5,25 cala oraz dyskiem twardym. Wszystkie te urządzenia są jednak elementami zewnętrznymi i wymagają stosowania specjalnych układów połączeń i obudów. Przewidziano również możliwość uzupełnienia pamięci operacyjnej do 640 KB.

Komputer w każdej wersji dostarczany jest użytkownikowi z pakietem oprogramowania. Pakiet ten stanowią dyskietki z systemem operacyjnym MS-DOS w wersji 3.3, cztery gry zręcznościowe oraz zintegrowany pakiet programowy o nazwie *First Choice*. Pakiet ten łączy w sobie edytor tekstu, bazę danych i arkusz obliczeniowy, zawiera pełną instrukcję obsługi wszystkich opcji zawartych w nim programów. *First Choice* przeznaczony jest dla początkujących użytkowników komputerów profesjonalnych, pozwala poznać terminologię stosowaną w profesjonalnych programach kalkulacyjnych i bazach danych oraz struktury tworzenia zbiorów dla tych programów. Pakiet zapewnia pełną wymienność tworzonych zbiorów.

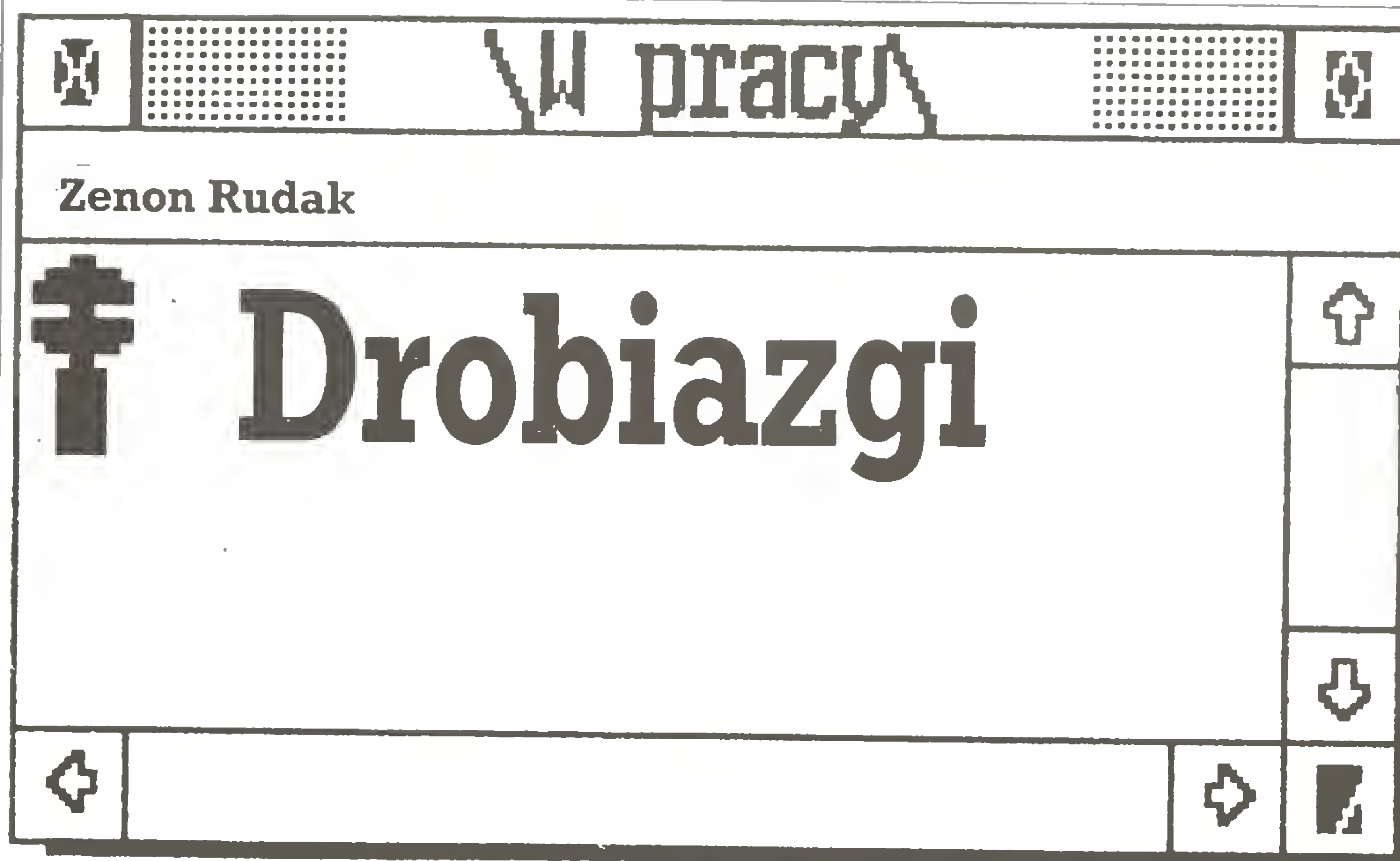


Wygląd całej maszyny w pełni odpowiada poziomowi włoskiego wzornictwa przemysłowego. Obudowa komputera jak i podstawa monitora zostały tak zaprojektowane, aby cały zestaw stanowił funkcjonalną bryłę, pozwalał użytkownikowi na dobrą organizację pracy i zajmował mało miejsca - komputer przeznaczony jest głównie dla osób pracujących w domu.

Olivetti oferuje na rynku angielskim komputer PC 1 w następujących konfiguracjach i cenach:

- wersja podstawowa z 1 napędem dyskowym i mono monitorem — 399 funtów;
- wersja pełna z dwoma napędami dyskowymi i kolorowym monitorem — 699 funtów;
- zewnętrzna stacja dyskietek 5,25 cala — 199 funtów;
- zewnętrzny dysk twardy 20 MB ze sterownikiem — 399 funtów.

Uzupełnienie pamięci operacyjnej wymaga dopłaty 69 funtów do ceny komputera.



Zenon Rudak

Drobiazgi

Na łamach naszego miesięcznika zainicjowaliśmy akcję poszukiwania producentów "galanterii" komputerowej. Do tej pory zgłosiło się do redakcji kilkunastu rzemieślników – producentów różnych drobiazgów wspomagających pracę z komputerem. W teście omawiamy kilka produktów dostarczonych do redakcji w czerwcu tego roku. Dziękujemy za to, co już nam przedstawiono i oczekujemy na dalsze propozycje oraz egzemplarze okazowe takich wyrobów. Akcja poszukiwania producentów trwa nadal.

Propozycja pierwsza - siatka

Siatka chroniąca oczy operatora komputerów przed promieniowaniem monitora dostarczona została do redakcji przez przedstawiciela Przedsiębiorstwa Innowacyjnego TETA, mieszczącego się we Wrocławiu przy ulicy Tenisowej 2C, tel. 675825.

Po raz pierwszy zauważyłem tego typu siatki na wystawie komputerowej w Londynie w ubiegłym roku. Wszyscy producenci oferujący je zdobywali natychmiast rzesze klientów. Napisałem o tym zjawisku w jednym z poprzednich numerów naszego pisma. Okazuje się, że i w Polsce nie brak ludzi przedsiębiorczych i energicznych. Spółka TETA uruchomiła produkcję takich siatek. Materiałem wyjściowym do produkcji jest importowana z Dalekiego Wschodu siatka nylonowa. Jest ona następnie barwiona na matowy czarny kolor i mocowana w dwukolorowej ramce z tworzyw sztucznych. Barwienie, produkcję ramki i montaż całości wykonuje spółka TETA. Wielkość ramki dobrana jest tak, że pozwala na wykorzystanie jej do monitorów 12- i 14-calowych. Barwienie siatki wykonane jest starannie. Nie widać zmian koloru na jej powierzchni użytkowej. Montaż siatki w ramce także jest prawidłowy. W egzemplarzach, jakie otrzymaliśmy, nie zauważa się skrzywienia osi nici siatki. Osie X i Y zachowują prostopadłość, co ma ogromny wpływ na jakość obrazu widzianego przez siatkę. Jej działanie polega na zmniejszeniu jaskrawości świecenia ekranu monitora przy zachowaniu kontrastu między punktami jasnymi i ciemnymi. Przedstawiona do testu siatka nie powodowała żadnych zniekształceń obrazu monitora monochromatycznego. Siatkę zakładałem na monitory o zielonej, bursztynowej i białej barwie ekranu. Zachowuje swoje działanie bez względu na rozdzielczość monitora. Szczególnie wyraźnie pozytywne działanie siatki zauważa się przy pracy z krajowym monitorem Neptun 156. Ma on bardzo jasną poświatę luminoforu i zastosowanie siatki znacznie łagodzi działanie jarzącego się ekranu na źrenice oczu przy wielogodzinnej pracy.

Zastosowanie siatki do monitora kolorowego powoduje powstawanie widocznych prążków interferencyjnych. Zjawisko to nasila się szczególnie, gdy ekran monitora jest dość wypukły lub gdy operator patrzy na monitor pod kątem. Zwiększenie odstępu siatki od powierzchni ekranu sytuację tę poprawia, ale jej nie likwiduje. Podobne zjawisko występuje, gdy siatkę umieści się przed ekranem telewizora kolorowego. Spółka TETA pracuje obecnie nad uruchomieniem produkcji siatek pozbawionych tej wady i przystosowanych do współpracy z monitorami kolorowymi.

Siatka mocowana jest do monitora za pomocą dwóch plastykowych uchwytów i "zatrzasków" typu "rzep". Zastosowane uchwyty pozwalają na ustalenie wymaganego dystansu między siatką a ekranem monitora. Mocowanie siatki polega na naklejeniu na obudowie monitora kawałków folii samoprzylepnej z "pętelkami" "rzepa". "Haczyki" "rzepa" znajdują się na końcach uchwytów. Po dobraniu (przez zaginanie uchwytu) odpowiedniego odstępu od szyby monitora wczepia się końce uchwytów w przyklejone do obudowy kawałki folii. Postępowanie takie jest bardzo proste i łatwe. Niestety nie jest pewne. Pod wpływem ciężaru siatki końce uchwytów wyczepiają się z pętelek i siatka spada. Wydaje się, że węzeł ten wymaga dopracowania (moim zdaniem należałoby zwiększyć długość zaczepu typu "rzep") lub zmienić koncepcję mocowania siatki do monitora.

Przedstawiona siatka jest produktem dobrym i niezwykle potrzebnym. Spółce TETA należy pogratulować szybkości działania i trafności wyboru produktu.


1

Propozycja druga - taśmy barwiące do drukarek

Taśmy barwiące do drukarek przedstawił pan mgr inż. Andrzej Franek prowadzący warsztat regeneracji kaset do drukarek komputerowych, ul. Sienkiewicza 6/21, 87-200 Wąbrzeźno, tel. 517.


3

Dostarczone do redakcji taśmy mają szerokość 13 mm i wykonane są z materiału, z jakiego wytwarza się taśmy do maszyn do pisania. Taśmy tego typu można stosować w drukarkach Star, Epson, NEC, Seikosha itp. Pan A. Franek konfekcjonuje je w odcinkach przeznaczonych do różnych drukarek, skleja i pakuje. Taśmy sklejane są w pętlę Mobiusa, co zapewnia całkowite wykorzystanie barwnika, jakim nasączona jest taśma. Końce taśm łączone są ze sobą za pomocą zgrzewania. Linia zgrzania jest cienka i nie powoduje przeszkód w druku. Jakość połączenia jest dobra i nie zauważa się pęknięć czy zerwania

> 40

39 <

taśmy przed wykorzystaniem barwnika. Odcinki taśm zwijane są w gęsty zygzak i pakowane w pudełka pozwalające na łatwe umieszczenie nowej taśmy w posiadanej kasecie. Pudełko z taśmą umieszczone jest w szczelnie zgrzewanej torebce foliowej zabezpieczającej barwnik przed wysychaniem. Producent poinformował o możliwości uruchomienia produkcji taśm o innej szerokości - przeznaczonych do drukarek Star serii LC, Amstrad, Atari itp.

Cena testowanych wkładów waha się w granicach 3500 - 4200 zł i zależy od długości odcinka taśmy. Proponowana cena wkładów porównywalna jest z ceną pobieraną przez punkty nasączające taśmy. Pan A. Franek realizuje dostawy wkładów taśm w sprzedaży wysyłkowej.

Propozycja trzecia - joystick MATT

Joystick MATT udostępnił redakcji do testowania przedstawiciel Pracowni Elektroniki Użytkowej i Przemysłowej, mgr inż. Tadeusz Trojak, Łódź ul. Wigury 15.

Joystick wykonany jest z czarnego tworzywa sztucznego. Rękojeść ma kształt ergonomiczny, przewidziano wgłębienia na palce i odpowiednio dobrano grubość uchwytu - tak, aby pasował do ręki. Na szczycie rękojeści umieszczono przycisk "strzał" w taki sposób, aby można było go łatwo naciskać kciukiem. Rękojeść mocowana jest za pomocą elastycznego przegubu do okrągłej podstawy. Podstawę zaopatrzono w cztery przyssawki gumowe pozwalające trwale umocować joystick do stołu. Taki sposób umocowania joysticka pozwala na energiczne posługiwanie się urządzeniem. Joystick połączony jest z komputerem za pomocą cienkiego przewodu wyposażonego we wtyczkę 9-stykową typu "D". Wtyczka wykonana jest z twardej gumy i stanowi integralną część przewodu połączeniowego. Można ją umieścić w gnieździe joystickowym każdego dostępnego na naszym rynku komputera. W poprzednio testowanych joystickach innych firm wtyczka sprawiała kłopoty i nie zawsze dawała się umieścić w gniazdach komputera Atari 800XL, 65XE, 130XE. Joystick MATT pozbawiony jest tej wady.

Joystick wykonany jest starannie. Pracuje poprawnie i na ogół bezawaryjnie. Redakcja otrzymała do testowania trzy takie joysticki. Po prawie czteromiesięcznej bardzo intensywnej eksploatacji (używał go młody entuzjasta gier komputerowych) jeden z joysticków przestał pracować. Pozostałe joysticki do dnia dzisiejszego pracują poprawnie*).

Joysticki MATT rozprowadzane są w sieci sklepów Centralnej Składnicy Harcerskiej. Cena joysticka wynosi 6750 zł i jest konkurencyjna z podobnymi wyrobami zagranicznymi oferowanymi w komisach i na giełdach komputerowych.

Propozycja czwarta - podstawka pod drukarkę

Podstawkę pod drukarkę typu PD-10 nadesłał do redakcji pan mgr inż. Andrzej Borysewicz, właściciel zakładu ślusarskiego mieszczącego się w Łodzi przy ulicy Skarżyńskiego 17. Podstawka PD-10 skonstruowana została w celu ułatwienia organizacji pracy i usytuowania drukarki współpracującej z komputerem. Przy dużej liczbie wydruków i zastosowaniu papieru we wstędze występują często kłopoty z prawidłowym ułożeniem papieru i bezawaryjnym odbieraniem gotowych wydruków z drukarki. Stosowanie taśmy papierowej wymaga pozostawienia wokół drukarki dużej ilości wolnego miejsca, o co w biurach czy naszych domach nie jest łatwo. Zadaniem przedstawionej podstawki PD-10 jest ułatwienie pracy drukarki przy druku na wstędze papierowej. Podstawkę wykonano z prętów stalowych i fornirowanej płyty drewnianej. Podstawka jest konstrukcją "piętrową". Ustawia się ją w miejscu, gdzie zazwyczaj stoi używana przez nas drukarka. Drukarkę ustawia się na "pierwszym piętrze" na płycie drewnianej. Zapas papieru we wstędze umieszcza się pod drukarką, między stalowymi prętami stanowiącymi nogi płyty utrzymującej drukarkę. W ten sposób drukarka wraz z papierem zajmuje miejsce przeznaczone poprzednio tylko dla drukarki. Aby zapewnić możliwość bezawaryjnego odbioru zadrukowanej taśmy papierowej, do płyty drewnianej podstawki montowany jest wykonany z giętego i spawanego drutu stalowego uchwyt będący magazynem zadrukowanego papieru. Uchwyt ukształtowany jest tak, że papier z drukarki "spływa" do magazynka bez zagnieceń i załamań. Uchwyt jak i podstawa zapewniają prawidłowe użytkowanie dostępnych na naszym rynku drukarek i dostępnego papieru we wstędze perforowanej. Producent przedstawił nam podstawkę PD-10 dla drukarek i papieru o szerokości 10 cali. W ulotce reklamowej dołączonej do wyrobu oferuje również takiej samej konstrukcji podstawkę dla drukarek i papieru o szerokości 15 cali. Przedstawiona podstawka wykonana jest starannie, choć nieodparcie odnosi się wrażenie dużej materiałochłonności wyrobu. Być może w następnych wersjach wrażenie to zostanie zatarte. Podstawkę pod drukarkę należy powitać na naszym rynku komputerowym z wielkim zadowoleniem. Tego typu meble są szansą zlikwidowania często istniejącego w wielu instytucjach widoku pracującej drukarki, do której papier wchodzi spod stołu, przesuwając się między różnymi kablami, koszem na śmiecie i telefonem, a wydruk spada na podłogę pod nogi interesantów. Są szansą na estetykę miejsca i komfort pracy.

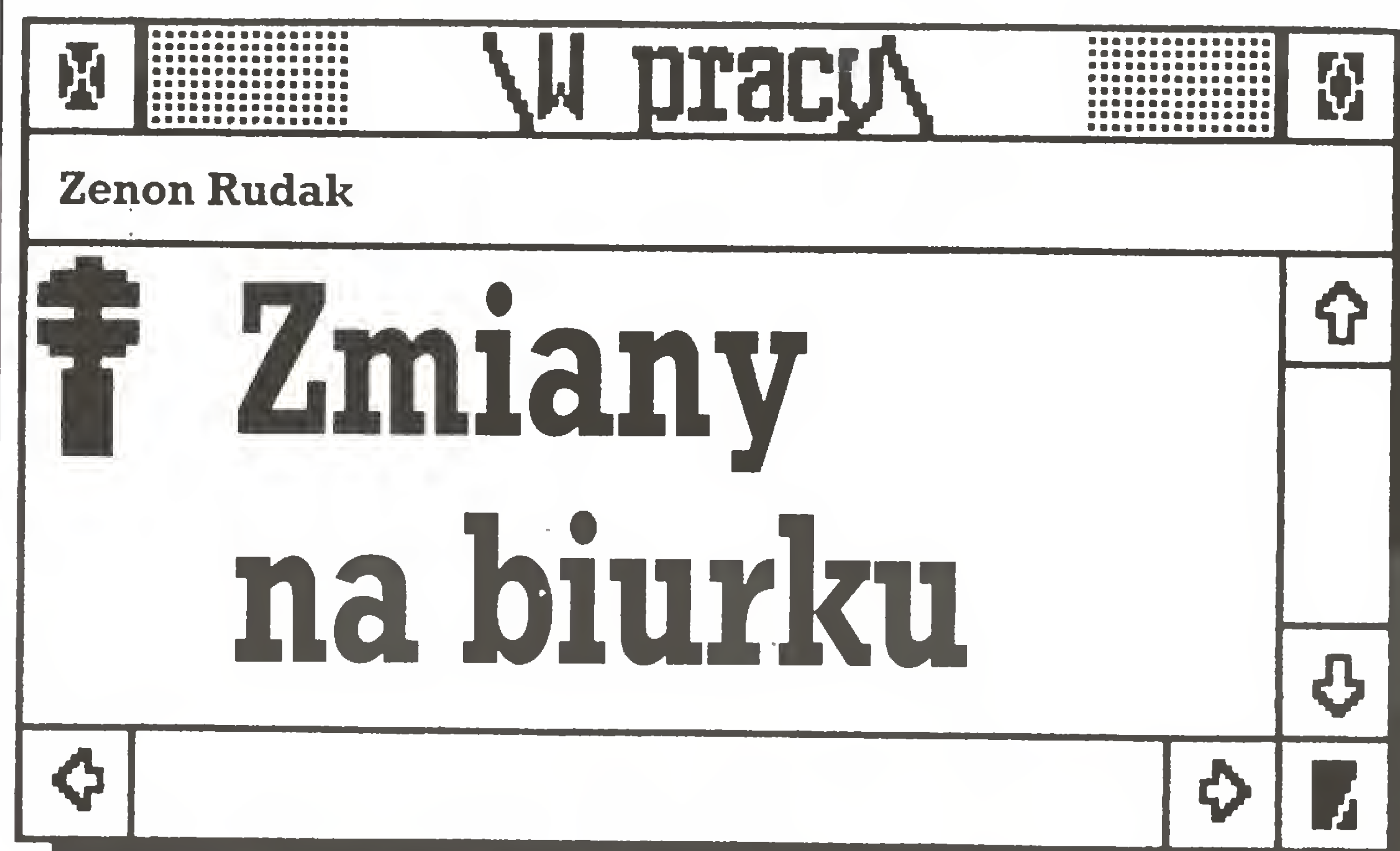
*) PRODUCENT WYJAŚNIA: Dostrzeżona wada wystąpiła w wyniku uszkodzenia przewodu. Obecnie stosowane są inne przewody – bardzo dobrej jakości.



4



3



Zenon Rudak

Zmiany na biurku

Gdy mówimy o profesjonalnym zastosowaniu komputerów w pracach biurowych, pracowniach projektowych, uczelniach, w zdecydowanej większości chodzi o komputery standardu IBM PC. Bardzo dobra polityka marketingowa i licencyjna niebieskiego koncernu sprawiła, że standard PC dominuje i jest jak na razie najlepszym rozwiązaniem dla pojedynczego stanowiska pracy. Pozycję standardu PC umacnia bardzo bogata biblioteka oprogramowania, pozwalająca rozwiązywać praktycznie dowolne problemy. Ale rozwijająca się cywilizacja wymaga przetwarzania coraz większych ilości informacji. Niedługo komputery standardu IBM PC nie będą mogły sprostać zadaniom. Nowe konstrukcje już powstają i zaczynają konkurować ze zmodernizowanymi komputerami IBM. Daje się zauważyć tendencje w rozwoju maszyn stanowiących pojedyncze stanowisko pracy (personal computer). Cechą podstawową nowych maszyn jest ich znacznie większa szybkość pracy w stosunku do komputerów IBM. Poprawiane są także parametry komunikacji z pamięcią zewnętrzną i ekranem graficznym.

Serce każdego komputera stanowi procesor. Tendencją ogólną nowych konstrukcji jest stosowanie nowych procesorów typu RISC (Reduced Instruction Set Computer). Procesory wykorzystywane obecnie posiadają bibliotekę składającą się z kilkudziesięciu lub kilkuset instrukcji. Czas wykonywania instrukcji wynosi od kilku do kilkunastu taktów zegara procesora. Pozwala to na wykonanie 2 - 4 milionów operacji na sekundę (MIPS). Procesor wyposażony jest także w kilka lub kilkanaście rejestrów pozwalających na przechowywanie danych i parametrów potrzebnych do realizacji instrukcji.

Procesory typu RISC są odmiennej konstrukcji. Biblioteka poleceń takiego procesora składa się z niewielkiej liczby (ok. 50) bardzo prostych instrukcji. Konstrukcyjnie procesor RISC pomyślany jest tak, aby każde polecenie możliwe było do wykonania w czasie od jednego do trzech taktów zegara taktującego. Ponadto procesory RISC wyposażone są w dużą liczbę rejestrów wewnętrznych (nawet do kilkuset), znacznie przyśpieszających przeszukiwanie danych, kontrolę programu, wykonywanie operacji logicznych i matematycznych. Są znacznie szybsze od najszybszych procesorów stosowanych obecnie i mogą bez trudu wykonywać 7 - 9 milionów operacji na sekundę (MIPS). Przy zastosowaniu najnowszych technologii półprzewodnikowych ich szybkość będzie mogła wynosić 20 - 40 MIPS.

Drugą tendencją w budowie nowych komputerów osobistych jest tworzenie systemów wieloprocesorowych. Idea ta polega na budowie bloków funkcjonalnych wyposażonych we własne systemy procesorowe. Procesor główny zarządza całym systemem zlecając wykonywanie np.: komunikacji z pamięcią, wyświetlanie obrazu, przesyłanie danych z i do pamięci zewnętrznej, wykonywanie obliczeń - układom wyposażonym w bardzo szybkie procesory specjalizowane. W czasie pracy bloku funkcyjnego procesor główny wykonuje dalszą część programu. Systemy tego typu pracują bardzo szybko i pozwalają na wykonywanie kilku programów jednocześnie.

Procesor każdego komputera współpracuje z pamięcią operacyjną i zewnętrzną. W nowych konstrukcjach stosuje się pamięci operacyjne o bardzo dużych pojemnościach rzędu 4 do 32 MB. Pamięć operacyjna zależnie od możliwości procesora adresowana jest ciągle lub dzielona na przełączalne obszary. Duże pojemności pamięci RAM konieczne są do pracy w systemach graficznych, do modelowania zjawisk fizycznych, do pracy programów optymalizujących konstrukcje techniczne.

Jako pamięć zewnętrzną komputery nowej generacji wykorzystywać mogą dyskietki elastyczne, dyski twarde, dyskietki optyczne, dyskietki typu CD-ROM, płyty wizyjne, kasety RAM. Rozwój sposobu magazynowania danych jest bardzo dynamiczny i trudno dokładnie przewidzieć, jakie rozwiązanie zdobędzie największą popularność. W wielu laboratoriach producentów sprzętu komputerowego przechodzą badania dyskietki elastyczne o średnicy 3,5 cala i możliwości zapisu ok. 10 MB informacji. Stacje takie wymagają jednak dyskietek o bardzo wysokiej jakości nośnika magnetycznego, na razie istnieje on tylko w laboratoriach. Stosowane obecnie dyski twarde posiadają zdolność zapisu do ok. 1 GB informacji. Od niedawna znane są metody zapisu dyskietek optycznych. Ich pojemność wynosi od 100 MB do kilku GB (giga bajtów). Systemy optyczne nie są jeszcze w pełni dopracowane. Istniejące rozwiązania rynkowe pozwalają na jednokrotny zapis i wielokrotny odczyt zapisanej informacji (WORM, Write Once - Read Many Times). Kasowanie polega na blokadzie fizycznego dostępu do zbioru. Skasowany zbiór pozostaje jednak nadal na nośniku i ogranicza pojemność dyskietki.

Prace nad nośnikami optycznymi są bardzo intensywne i rozwiązanie problemu wielokrotnego zapisu i odczytu takiego nośnika jest kwestią czasu i zaangażowania większych środków finansowych. Do odczytu danych wykorzystywanych wielokrotnie przewiduje się stosowanie dyskietek zapisywanych podobnie jak płyty typu compact disk - CD-ROM. Ta metoda zapisu jest już doskonale opanowana, jest niezawodna i tania. Pojemność dyskietki typu compact wynosi 1 - 8 GB. Przewiduje się wyposażanie nowych komputerów w interfejsy pozwalające wykorzystywać domowe urządzenia do odtwarzania płyt compactowych jako stacji wprowadzania danych lub programów. Dyskietki typu compactowego mogą służyć tylko do odczytu programów lub danych. System ten nie umożliwia zapisu. Zaletą systemu CD-ROM jest brak możliwości nielegalnego kopiowania zawartości takiego nośnika.

Pamięci zewnętrzne z płytą compactową zastosowano już w systemach sieci komputerowych szkolnictwa angielskiego. Przystawki do odczytu płyt laserowych sterowane są przez komputery



Acorn Master. Na płytach CD-ROM zapisywane są dane encyklopedyczne, wybitne dzieła literackie, dane kartograficzne.

Taniejące technologie i materiały półprzewodnikowe stworzyły możliwości stosowania jako pamięci zewnętrznej kaset RAM. Kaseta taka ma wielkość karty kredytowej i w procesie produkcji zostaje na niej umieszczony obszar pamięci RAM wraz z baterią zasilającą. Bateria podtrzymuje działanie układów pamięci przez ok. 5 lat bez obaw o utratę zapisanych informacji. Kaseta może być dołączona do komputera i wykorzystywana jak normalna dyskietka. Możliwy jest swobodny zapis i kasowanie zbiorów. Odłączenie kasety od komputera nie powoduje utraty zapisanych informacji. Prowadzi się próby z kasetami o pojemności od 1 MB do 10 MB. Zaletą kaset RAM jest bardzo szybki odczyt i zapis informacji (kaseta po

> 42

41 <

przyłączeniu do komputera zachowuje się jak duży RAM-dysk), wyeliminowanie specjalnych stacji z układami napędowymi i odczytującymi zawartość nośnika. Pewną wadą jest ograniczony czas przechowywania informacji.

Niezwykle ważnym elementem komunikacji użytkownika z komputerem jest monitor graficzny. Budowa monitorów i układów sterujących grafiką zmienia się wraz ze zmianami samych komputerów. Stosowanie wysokiej jakości programów graficznych, a w szczególności programów do edycji tekstów typu pulpit wydawniczy (DTP), wymusiło powstanie specjalnych sterowników zapewniających wysoką rozdzielczość obrazu. Sterowniki te stanowią obecnie uzupełnienie lub dodatkowe wyposażenie istniejących komputerów. Programy typu DTP wymagają rozdzielczości obrazu co najmniej 1024 na 1024 punkty. Przewiduje się, że w nowo powstających konstrukcjach rozdzielczość ekranu wynosić będzie 2048 na 2048 punktów. Rozdzielczość ta ma zapewnić stosowanie zasady WYSWYG ("widzisz jak będzie wydrukowane") przy pracy z programami graficznymi.

Do prac graficznych konstruktorzy przewidują stosowanie monitorów o przekątnej ekranu od 16 do 19 cali. Monitory barwne mają zapewnić wyświetlanie jednocześnie do 4096 kolorów z palety 16 milionów kolorów. Liczba kolorów wyświetlanego obrazu zależna



jest od przyjętego systemu opisu bajtowego i pojemności pamięci współpracującej z procesorem obrazu. W zastosowaniach specjalnych podane wyżej wielkości mogą być jeszcze powiększone. Do prac biurowych przewiduje się stosowanie monitorów o przekątnej ekranu od 14 do 16 cali. Będą stosowane monitory monochromatyczne i kolorowe. Sterowniki ekranowe nowych komputerów przystosowane prawdopodobnie zostaną do przyjmowania i przetwarzania obrazu generowanego także poza komputerem np. z kamery wideo, odtwarzacza laserowych płyt wizyjnych lub magnetowidu. Systemy tego typu pozwolą na rozszerzenie możliwości graficznych komputera a jednocześnie będą pomocne przy modelowaniu zjawisk fizycznych, analizie obrazów mikroskopowych, kontroli szybkozmiennych procesów.

Opisane wyżej przewidywane zmiany konstrukcyjne komputerów osobistych mają przyśpieszyć i ułatwiać pracę naukowcom, konstruktorom, nauczycielom. Przewiduje się większy niż obecnie udział tych maszyn w sterowaniu aparaturą kontrolno-pomiarową i eksperymentalną. Możliwość zapisu i szybkiego dostępu do znacznej ilości danych ma ułatwiać procesy analityczne i decyzyjne. Powstające komputery mają pełnić rolę pierwszych ogniw złożonego procesu powstawania sztucznej inteligencji.

Przedstawione w tym artykule informacje pochodzą z dyskusji, jaka przeprowadzona została w gronie informatyków i menagerów światowego rynku komputerowego, zamieszczonej na łamach angielskiego miesięcznika "Personal Computer World" w 1988 roku.

W pracy

Prosto z dysku

PagePerfect 1.0

Powoli zbliżają się ku sobie światy programów redagujących (procesorów tekstu) i edytorskich (desktop publishing). Z początku funkcje ich były całkowicie rozłączne: klasyczny *WordStar 3.0* nie pozwala praktycznie na żadne działania kształtujące postać uzyskanego dokumentu, natomiast pierwsze wersje *Ventury* i *PageMakera* praktycznie uniemożliwiały bezpośrednie wprowadzenie jakiegokolwiek dłuższego tekstu. Obecnie najnowsze wersje programów redagujących *(WordPerfect 5.0; MS Word 4.0)* zawierają całkiem rozbudowane funkcje kształtowania strony.

Pojawiło się więc miejsce na nowy rodzaj pakietu zintegrowanego: program umożliwiający poprawne (choć bez wielu najbardziej zaawansowanych funkcji) redagowanie i opracowanie strony, włączanie rysunków itp. Takim właśnie programem jest *PagePerfect 1.0* wprowadzony na rynek przez firmę **International Microcomputer Software.** Podobieństwo nazwy do programu *WordPerfect* nie oznacza nic poza znaną z ciekawostek biologicznych skłonnością walczących o byt gatunków do stosowania mimikry.

Grammatik III

Programy, dumnie zwane redagującymi lub wręcz procesorami tekstu, w istocie wciąż oferują użytkownikowi niewiele ponad rozszerzoną wersję funkcji zwykłej maszyny do pisania: pozwalają łatwo wprowadzać i poprawiać tekst oraz dbać o jego formę, ignorując jednak całkowicie treść. Jak dotąd szczytem dostępnej dla użytkownika PC analizy sensu wklepywanych liter była kontrola ortograficznej zgodności pisowni ze słownikowym wzorcem. Obecnie autorzy tekstów angielskojęzycznych mogą za niecałe 100$ sprawdzić także ich poprawność gramatyczną. Jest to milowy krok w kierunku poprawy inteligencji programów redagujących.

Program *Grammatik III* firmy **Reference Software** wykrywa, sygnalizuje i sugeruje sposób poprawy powtarzających się słów, błędnej interpunkcji, błędnego użycia strony biernej, powtarzającego się używania fraz pretensjonalnych i kalek językowych, podwójnych zaprzeczeń, niepełnych zdań lub niezgodności podmiotu z orzeczeniem. Przykład: natrafiając na zdanie *"Their going to have a party"* program sugeruje: *"Kontekst 'their' wskazuje, że zapewne powinno być 'there' lub 'they're'"*.

Niestety - tego programu nie da się łatwo przerobić na polski.

Tornado

Nie brak nam programów pozwalających skutecznie administrować bazą danych zawierającą setki i tysiące najdziwniej rozbudowanych i powiązanych ze sobą zapisów, na co dzień jednak borykamy się zwykle beznadziejnie ze znacznie prostszym problemem: jak poradzić sobie z zalewem krótkich, słabo ze sobą powiązanych notatek zapełniających nasze dyski. *Tornado* to przedstawiona przez firmę ***Micro Logic* Corp.** propozycja rozwiązania tego problemu - rezydentny program zajmujący 57 KB pamięci pozwala wprowadzić jakikolwiek porządek w stercie 25.000 notatek pogrupowanych w 50 "pakietów".

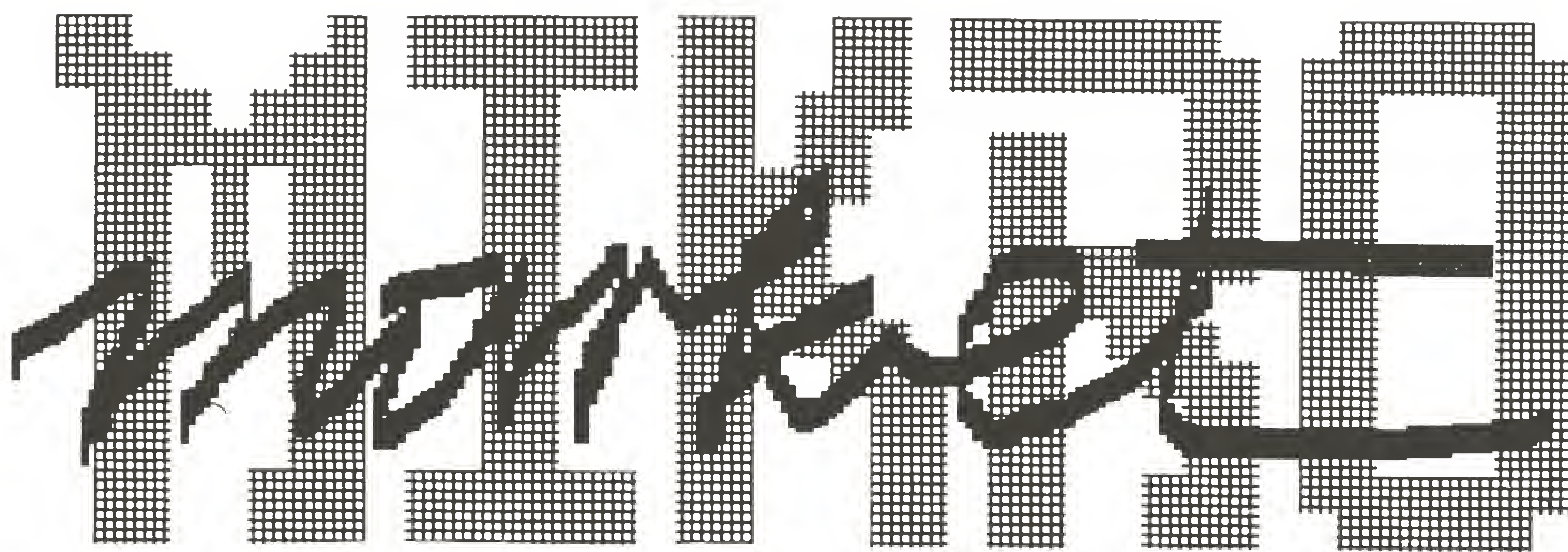
Jak to się sprawdzi w życiu - życie pokaże.

Grandview

To nieco droższy i obszerniejszy program o podobnej funkcji, którego twórcy z firmy **Symantec Corp.** wieszczą powstanie nowej kategorii programów: personal information menager, czyli organizator notatnika.

Najbardziej znaną zapowiedzią z tej kategorii jest program *Lotus Agenda*, którego faktyczne wprowadzenie na rynek wciąż się opóźnia.

opr. W. Majewski



Sprzedam spółkę!

Z TADEUSZEM BROWARKIEM, prezesem spółki "Lumena", rozmawia Władysław Majewski.

"Sprzedam udział w spółce!" - ćwierćstronicowe ogłoszenie w jesiennych numerach "Życia Warszawy" wzbudziło pod koniec 1987 r. spore zainteresowanie...

Byłeś drugim dziennikarzem, który zapytał mnie o przyczyny tej decyzji oraz o odzew, jaki wywołała oferta. Pierwszy był "Financial Times"...

Komu w końcu sprzedałeś? Chętnych było wielu...

Byłym wspólnikom.

To trzeba się było ogłaszać? Sporo wysoko opodatkowanych pieniędzy wydałeś z własnej kieszeni!

Ogłoszenie było środkiem nacisku na wspólników. Zgodnie z umową spółki w ciągu miesiąca od pisemnego powiadomienia o zamiarze sprzedaży udziału mieli oni prawo do pierwokupu. Dokładnie od podanego w ogłoszeniu dnia uzyskiwałem prawo do sprzedaży dowolnej wybranej przez siebie osobie oraz ujawnienia każdemu potencjalnemu nabywcy stanu finansów spółki: danych o obrocie, zysku, kapitałach i prowadzonych przez "Numericę" interesach.

Poskutkowało?

Tak. Umowę sprzedaży podpisaliśmy na dwie godziny przed upływem terminu, po kilkunastogodzinnym, nerwowym maratonie...

W takim razie Wasze interesy musiały być mocno podejrzane, skoro prawda o nich była tak groźna...

Fiskusowi są one doskonale znane, ujawniać ich natomiast konkurencji lub prasie moi partnerzy nie mieli ochoty. Ogłoszenie posiadało też zarazem i drugą funkcję: miał przekonać ich o stanowczości mojej decyzji. Przez pierwsze dni po zawiadomieniu sądzili, że tylko się droczę.

Rozmawiamy stale o zewnętrznych przejawach tej transakcji, bardzo zresztą, jak na nasze zwyczaje, barwnych. Czy możesz jednak teraz, z perspektywy paru miesięcy, powiedzieć, po co sprzedałeś 1/3 znakomicie prosperującej spółki?

By założyć nową.

Nie rozumiem. Grupa młodych, wysoko kwalifikowanych i od dawna zaprzyjaźnionych fachowców wielkim wysiłkiem zakłada firmę, która w ciągu kilku kolejnych miesięcy równie znojnych starań uzyskuje wielomilionowe obroty, liczne grono bardzo wymagających, ale i wysoko ceniących wyniki zawartych transakcji klientów, własne interesujące opracowania techniczne - by po roku całą tę drogę przechodzić od nowa, pozostawić zdobyte dobre imię, klientów, lokale, warsztat pracy i zgrany zespół?

Mógłbym odpowiedzieć tak: co własne, to własne, a w "Lumenie" mam udział znacznie większy niż w "Numerice".

Chyba unikasz istoty sprawy. Takie motywy mogły kierować Tobą, ale nie grupą Twoich współpracowników, którzy również przeszli do "Lumeny".

To prawda. Spójrzmy więc na to inaczej: półtora roku temu nie mieliśmy nic i żadnych problemów, jak to wykorzystać. Nadszedł jednak moment, w którym trzeba było podjąć decyzję, jak wykorzystać zgromadzony potencjał ludzki, finansowy i materialny, w jakim kierunku się rozwijać. Ja wraz z grupą pracowników mieliśmy jeden pomysł, pozostali wspólnicy - inny. Trzeba było się rozejść, zanim się do końca nawzajem znielubimy.

I komu życie przyznało rację?

Nikomu. "Lumena" w pół roku od swego powstania ma obroty zbliżone do kolegów z dawnej spółki (o ile mogę je szacować). Wyspecjalizowaliśmy się w dostawach wysokiej klasy sprzętu pomiarowego oraz komputerowych systemów zbierania danych i sterowania eksperymentem naukowym. Naszymi klientami są głównie najbardziej wymagające placówki naukowe, a wśród naszych dostawców, którzy powierzyli nam prawo serwisu i reprezentacji na polskim rynku, znajdują się czołowi światowi producenci tego typu urządzeń.

Na wizytówce czytam: Tadeusz Browarek, a cały świat studenckiej turystyki od lat zna Tomka z Almaturu...

Tak przedstawiłem się na balu maturalnym dziewczynie, która została później moją pierwszą żoną. Od niej nauczyli się tak mówić do mnie koledzy i zostało do dziś.

Co studiowałeś?

Ukończyłem Wydział Fizyki Technicznej i Matematyki Stosowanej Politechniki Warszawskiej, potem pracowałem przez wiele lat w Instytucie Chemii Fizycznej PAN. Widziałem tam naukowców tak zabieganych za chałturami, że nie starczało im już czasu na ambicje naukowe. To doświadczenie wiele mnie nauczyło: nie chciałem stać się jeszcze jednym z tego kręgu pracy pozornej.

Potem pracowałem w Ośrodku Badawczo-Rozwojowym Informatyki, gdzie poznałem tajniki obrotu oprogramowaniem i systemem odgórnej komputeryzacji za państwowe pieniądze, jaki funkcjonował u nas do niedawna. Instalowałem m.in. we współpracujących firmach system RODAN.

A w czym pomogły Ci w założeniu własnej spółki doświadczenia z Almaturu?

Była to znakomita szkoła pracy nie limitowanej godzinami, no i kopalnia kontaktów: wszyscy dzisiejsi pracownicy placówek naukowych niegdyś byli studentami, a wielu wędrowało z "Almaturem".

Co właściwie im sprzedajesz, za co bierzesz pieniądze od placówek naukowych, doskonale znających swój sprzęt i potrzeby? Czy nie zarabiasz na luce prawnej, dzięki której uczelni łatwiej kupić drożej od spółki takiej jak Twoja niż bezpośrednio od producenta?

Nasza firma pełni kilka ważnych ról: jesteśmy filtrem, nie dopuszczającym złego sprzętu do końcowego użytkownika oraz buforem, kumulującym fundusz gwarancyjny i znakomicie podnoszącym stan realnej gotowości technicznej sprzętu, jesteśmy zbiornicą informacji techniczno-handlowej, która jest często trudno dostępna. Służymy naszym klientom bezpłatnym doradztwem w zakresie oceny jego potrzeb i możliwości ich zaspokojenia, co w przypadku sprzętu wysokospecjalistycznego nie jest wiedzą łatwo dostępną i pozwala często uniknąć mozolnej budowy specjalistycznego sprzętu do jednorazowego użytku w sytuacjach, gdy można zastosować rozwiązania typowe.

Niebanalnym wreszcie naszym wkładem jest oprogramowanie dostarczanych systemów zbierania danych, np. opracowany przez nas system akwizycji i opracowania danych pomiarowych silników spalinowych przynosi już gospodarce realne korzyści.

Słyszałem, że zakładanie spółek weszło Ci w krew...

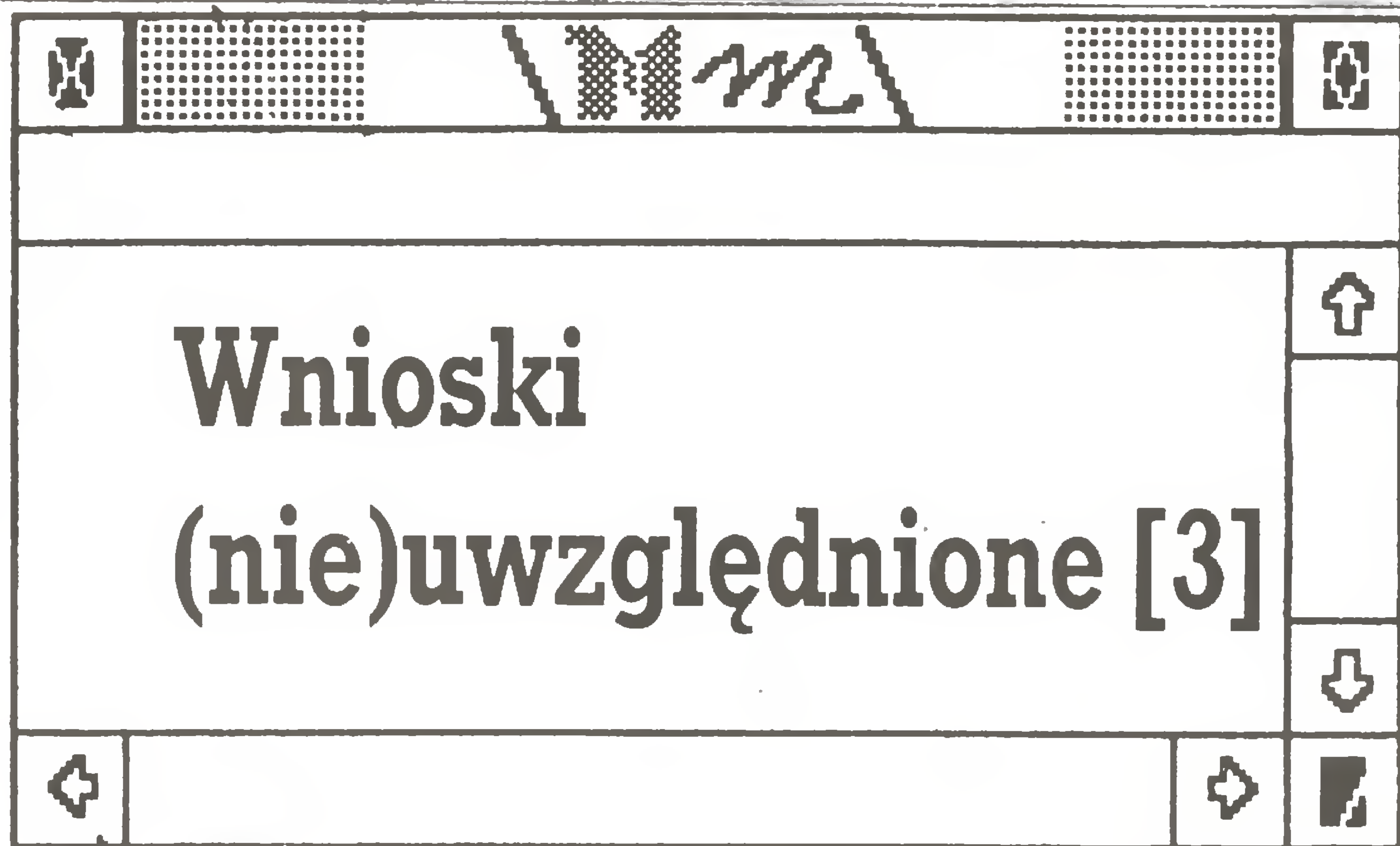
Tak, sprzedajemy skutecznie swoją wiedzę praktyczną w tej dziedzinie, doradzamy i załatwiamy wiele spraw związanych z rejestracją spółki i uzyskaniem przez nią zezwolenia na działalność gospodarczą. Mieliśmy już ok. 10 klientów, nieraz ułatwiamy start potencjalnej konkurencji.

To w końcu jesteś inżynierem czy adwokatem?

Korzystamy z luki rynkowej, jaką jest brak w Polsce firm prawniczych działających w sferze gospodarczo-prawnej. A co do kwalifikacji: zatrudniam również prawników i ekonomistów.

Słyszałem również, że jesteś filantropem?

Polski Związek Lekkiej Atletyki, który wynajmuje sąsiedni lokal, poprosił nas o wsparcie. Okazuje się, że dotacje moglibyśmy wypłacić tylko z czystego, opodatkowanego zysku, a chyba inaczej powinny być opodatkowane środki, których spółka ani udziałowcy nie skonsumowali (tłumacząc na polski, od każdej złotówki dotacji dwie należałoby odprowadzić do Urzędu Skarbowego). PZLA sprawę przedstawił w "Sztandarze Młodych", który następnie poprosił mnie o wypowiedź. W istocie nie wsparłem jeszcze nikogo, a chętnie pomógłbym towarzystwu rodziców dzieci bezglutenowych.



Produkcja oprogramowania stawia także przed władzami podatkowymi wiele zupełnie nowych problemów. Próbują one pojęcia dostosowane do drobnej produkcji materialnej przenieść w obszar wysoko kapitałochłonnej i błyskawicznie starzejącej się twórczości intelektualnej. Prowadzi to do wielu nieporozumień, czasem zabawnych, lecz nie dla zainteresowanego. W rezultacie wiele firm prowadzi obfitą korespondencję, starając się wyjaśnić o co w tym interesie chodzi. Rezultaty bywają różne.

Kilka firm pozwoliło nam zapoznać się z wnioskami kierowanymi do władz finansowych oraz z nadesłanymi przez resort finansów uzasadnieniami - najczęściej odmownych - decyzji. Mogą one być dobrym wprowadzeniem do dyskusji o warunkach prawnych produkcji oprogramowania oraz do formułowania planów dalszego działania.

Pisma te pogrupowaliśmy według spraw, których dotyczą. Poszczególne problemy omawiać będziemy w kolejnych numerach. Z wielu podobnych wniosków wybraliśmy najistotniejsze argumenty o znaczeniu ogólniejszym. Prosimy równocześnie o nadsyłanie nam podobnych spraw i korespondencji.

Odsłona 3: Czy program się zużywa?

Wniosek (do departamentu planowania i finansowania w Urzędzie Postępu Naukowo-Technicznego i Wdrożeń):

Proszę o zajęcie stanowiska w sprawie zmiany przepisów prawno-finansowych dotyczących kwalifikacji oprogramowania jako czynnika rzeczowego warunkującego działalność produkcyjną przedsiębiorstw komputerowych. Obowiązujące przepisy stanowią barierę trudniejszą do pokonania od embarga krajów zachodnich.

Producenci oprogramowania zaangażują się poważnie w programowanie mikrokomputerów 32-bitowych, gdy liczba sprzedanych egzemplarzy zapewni im opłacalność własnych produktów, czyli oprogramowania powielarnego. Producent musi sam się finansować i ponosi całkowite ryzyko za podejmowane decyzje co do kierunków i rozmiarów produkcji. Tymczasem obecne przepisy nie umożliwiają reprodukcji posiadanych środków produkcji.

Uzasadnienie:

Naszym produktem są programy, które muszą mieć walory użytkowe umożliwiające ich profesjonalne zastosowanie zgodne ze standardem światowym.

Wymaga to korzystania ze światowego zasobu wiedzy (zakupu nowoczesnej technologii komputerowej, oprogramowania narzędziowego itp.), czyli zakupu wiedzy (know-how), co jest kapitałochłonne (do kilkudziesięciu mln zł za program). Wiedza ta ma zastosowanie w ściśle określonym czasie (ze względu na szybkie zmiany technologii), dla konkretnego typu sprzętu i środowiska programowego, dla konkretnej technologii przetwarzania oraz dla konkretnego zakresu użytkowego wytworzonego programu.

Zmiana technologii powoduje bezużyteczność wyprodukowanych programów, co zmusza do stałej intensyfikacji prac - czas produkcji programu nie może być dłuższy od okresu zmian technologicznych.

Przedmiotem kalkulacji powinien być zatem każdy program jednoznacznie określony jako produkt z osobna.

(...) Traktowanie oprogramowania jako wyposażenia mikrokomputera nie budzi zastrzeżeń w przypadku systemów operacyjnych, asemblerów czy kompilatorów. Pozostałe oprogramowanie należy zakwalifikować jako materiały lub licencje, gdyż są przydatne praktycznie jednorazowo - jako wzór. (...)

Proponujemy więc, by każdy zakupiony program zaliczany był przez technologa-informatyka do jednej z czterech grup:

1) wyposażenie
2) narzędzia
3) materiał
4) licencja

Programy zakwalifikowane jako WYPOSAŻENIE powinny być rozliczane w postaci odpisów, których okres powinien wynosić 3-5 lat, a nie 7-10, jak to jest praktykowane.

Programy zakwalifikowane jako NARZĘDZIA (przedmiot nietrwały) powinny być rozliczane w ciągu 2-3 lat - gdyż tyle zwykle trwa dana generacja sprzętu, systemów operacyjnych lub kompilatorów.

Programy zakwalifikowane jako MATERIAŁ należy rozliczać w okresie trwania otwartego zlecenia na produkcję danego programu (1-2 lata) w formie rozliczeń międzyokresowych, gdzie okres rozliczeń ustala się z góry.

Programy zakupione w formie LICENCJI (koszty przygotowania nowej produkcji) należy rozliczać tak jak zakwalifikowane jako materiał. W koszty kopii programu należy wliczyć odpowiednią do zaplanowanej wielkości sprzedaży część kosztów zakupu licencji. Przy sprzedaży większej liczby kopii programu koszty przygotowania produkcji nie będą naliczane.

Odpowiedź nie jest nam znana

Glosa redaktora:

Program (zwłaszcza w postaci źródłowej) wraz z dokumentacją, jako mienie wielkiej wartości (często kilku milionów złotych) nie mające postaci materialnej lub trwale z nim związanego nośnika materialnego i nie ulegający w głowie korzystającego zeń widocznemu zużyciu nie mieści się w systemie pojęć organów finansowych, stąd najbardziej karkołomne interpretacje spotykane w praktyce.

Najczęściej kwestionuje się całkowicie przydatność oprogramowania wzorcowego i narzędziowego w procesie produkcji nowych programów lub traktuje się oprogramowanie jako nie ulegający widocznemu zużyciu środek trwały, o okresie amortyzacji bliskim 10 lat.

Jest to obecnie najbardziej dokuczliwa bariera rozwoju branży informatycznej.

Odsłona 4: Czy program to dzieło?

Wniosek:

Wnoszę o wyrażenie zgody na opodatkowanie osób fizycznych świadczących na rzecz naszej firmy pracę z tytułu umów o dzieło podatkiem od wynagrodzeń, zamiast obrotowym i dochodowym.

Uzasadnienie:

Przedmiot naszej działalności gospodarczej wymaga korzystania z umiejętności fachowych z wielu dyscyplin wiedzy. Uzasadnia to zawieranie umów o dzieło lub zlecenia z osobami nie zatrudnionymi w zakładzie. Dla przykładu prace nad finansowo-księgowym systemem ekspertowym wymagają okresowej intensywnej współpracy z ekonomistami, specjalistami od modelowania procesów gospodarczych oraz prawnikami; prace nad systemem analizy tekstów wymagają współpracy ze specjalistami z zakresu językoznawstwa, logiki, psychologii kognitywnej itp; w tworzeniu systemu ekspertowego wspomagania pracy lekarza konieczna jest pomoc specjalistów od farmakologii klinicznej, chemii leków, medycyny ogólnej, inżynierii biomedycznej itp.

Korzystanie ze świadczeń tych osób jest jednak znacznie utrudnione ze względu na objęcie ich dochodów podatkiem obrotowym i dochodowym. Jakkolwiek osoby te nie prowadzą stałej działalności uzasadniającej pobranie podatku obrotowego, a uzyskany dochód mieści się z reguły w granicach wolnych od podatku, to spoczywa na nich obowiązek odpowiednich zgłoszeń i deklaracji. Ponieważ większość obywateli nigdy nie uczestniczyła osobiście w postępowaniu podatkowym w obawie przed niewiadomymi trudnościami rezygnują oni z dodatkowego źródła zarobków.

Uwzględnienie wniosku jest również zgodne z interesem Skarbu Państwa, gdyż należy się spodziewać, że wpływy z tytułu opodatkowania świadczeń wypłaconych przez naszą firmę na rzecz usługodawców wzrosną kilkakrotnie, wnioskodawcy natomiast ułatwi zawieranie niezbędnych umów o dzieło lub zlecenia.

Decyzja (marzec 88):

Uwzględniając przedstawione motywy przychody wymienionych osób poddaje się podatkowi od wynagrodzeń w wysokości 21,5% od każdej wypłaty po potrąceniu 20% na koszty uzyskania pod warunkiem, że wypłaty dokonywane będą osobom pozostającym w stosunku pracy w jednostkach gospodarki uspołecznionej i przedmiotem umów będą usługi wykonywane osobiście i nie wchodzące w zakres działalności, od której osoba zawierająca umowę o dzieło lub zlecenie obowiązana jest opłacać podatek obrotowy.

Do poboru podatku od wynagrodzeń od wymienionych osób obowiązany jest jako płatnik Wasz zakład, który również zobowiązuje się do przekazywania informacji jednostkom gospodarki uspołecznionej zatrudniającym zleceniobiorców o wysokości dokonanych z tego tytułu wypłat.

Decyzja obowiązuje do czasu nowego prawnego uregulowania powyższej kwestii.

Glosa redaktora:

Izba okazała się łaskawa, gdy widoczne są natychmiastowe dodatkowe wpływy i możliwość przerzucenia niemałych obowiązków i odpowiedzialności na firmę prywatną. Decyzja ta ma jednak charakter jednostkowy (dotyczy tylko osób korzystających z usług wnioskujących firm) i opatrzona jest dość dziwnym w sensie prawnym zastrzeżeniem zawartym w ostatnim zdaniu.

Od 1 lipca br. zasada ta została wprowadzona powszechnie.

opracował
Władysław Majewski



Zamierzacie Państwo wprowadzić mikrokomputery do Waszego Zakładu? Wybierzcie właściwego partnera!

Nasza oferta obejmuje:

Produkcję mikrokomputerów ALMA XT/AT
Doradztwo
Instalacje systemów i sieci
Opracowywanie i wdrażanie oprogramowania
Szkolenie
Gwarancje
Serwis pogwarancyjny

WYKONAMY OPROGRAMOWANIE UŻYTKOWE ZGODNE Z PAŃSTWA POTRZEBAMI.
w tym w szczególności systemów płacowych, środków trwałych, finansowo księgowe, magazynowe i wspomagania prac biurowych.

Ko-7

COMBIT

Przedsiębiorstwo Techniczno-Handlowe Sp. z o.o.

Katowice ul. Nasypowa 6, tel. 521-300

poleca

poprzez swoje zakłady kompleksowe usługi w zakresie:

- dostaw sprzętu elektronicznego i mikrokomputerowego
- oprogramowania systemów informatycznych
- zabezpieczenia elektrostatycznego

Oferta sprzętowa obejmuje:

- zestawy mikrokomputerowe w dowolnej konfiguracji
- urządzenia elektroniki profesjonalnej (zestawy pomiarowe, aparaturę medyczną, telefaxy, zasilacze, urządzenia ochrony przed przepięciami)
- urządzenia małej poligrafii

a także:

- aparaturę audio-wideo oraz zestawy do odbioru telewizji satelitarnej.

Oferujemy również:

- bogatą gamę programów użytkowych, zarówno gotowych jak i realizowanych na indywidualne zamówienie

Ponadto zapewniamy:

- pełną ochronę elektrostatyczną stanowisk komputerowych, stanowisk montażu układów MOS, pomieszczeń biurowych, hal produkcyjno-montażowych.

Polecamy indywidualne środki ochrony elektrostatycznej

- pojemniki
- ubrania
- wykładziny

Jeżeli chcesz sprostać wymogom nowoczesności, powierz realizację tego zamierzenia
Przedsiębiorstwu Techniczno--Handlowemu "COMBIT" w Katowicach ul. Nasypowa 6, tel. 521-300

Ko-134/146-7

WPHW O/Dąbrowa Górnicza
Sklep nr 163 "ELEKTRON"

41-300 Dąbrowa Górnicza
ul. Sobieskiego 17

Jeden z pierwszych i najtańszych uspołecznionych punktów sprzedaży sprzętu minikomputerowego prowadzi skup i sprzedaż:

- **minikomputerów 8 - bitowych**

(ceny sprzedaży)

np. Spectrum 128K +2 - 400.000.-
Atari 65 XE + mag. - 350.000.-
130 XE + mag. - 420.000.-
520 ST + SM 124 - 1.800.000.-
Amstrad CPC464 z ziel.mon . - 520.000.-
CPC6128 z ziel.mon. - 950.000.-
Commodore C128 - 500.000.-

- **drukarki (z kablami)**

np. DMP 2000 - 650.000.-
STAR NL 10 - 830.000.-
SG 15 - 1.150.000.-
NX 15 - 1.250.000.-

- **systemy minikomputerowe PC XT/AT w dowolnej konfiguracji**
- **urządzenia peryferyjne**
- **sprzęt wideo i CTV**

np. magnetowid SANYO 3100EE - 980.000.-
JVC HR-D210EE - 1.150.000.-
telewizor SANYO 21" - 1.150.000.-
SANYO 26" - 1.650.000.-
Panasonic 26" - 1.750.000.-
kamera Panasonic M5 - 4.100.000.-

- **osprzęt i wyposażenie:** dyskietki, kable, kasety, pojemniki, elementy elektroniczne, urządzenia dodatkowe.

Sklep prowadzi sprzedaż pozarynkową.

Udziela gwarancji i zapewnia serwis poprzez PHT "ABM" Sp. z o.o. w Dąbrowie Górniczej ul. Czerwonych Sztandarów 94.

Zapraszamy do sklepu w godz. 10 - 18.

Dąbrowa Górnicza ul. Sobieskiego 17 tel. 62-23-71

Ko-129/149/10

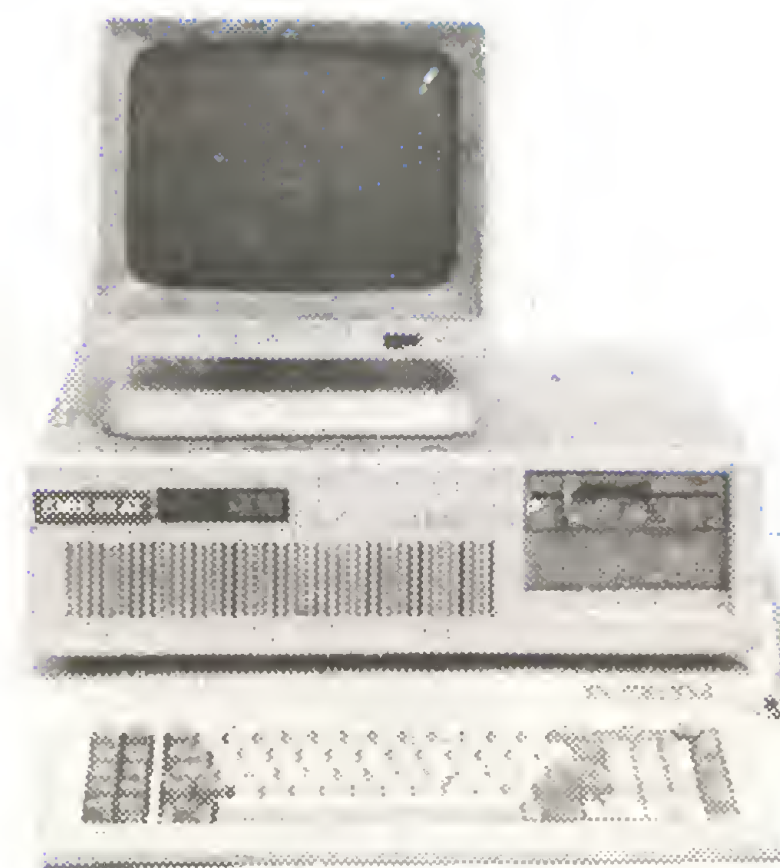
PRZEDSIĘBIORSTWO ZAGRANICZNE WIELOBRANŻOWE "EMIX"



HANNA KUBIAK
Biuro Techniczne i Informacyjno-Handlowe
ul. Smoleńskiego 4 m.17-18 01-698 WARSZAWA
TEL. 33-57-36, 33-10-85 TLX 815871 emix pl

EMIX 86 XT Turbo

- pamięć RAM 640 KB
- zegar 4,77/8 MHz
- 2 jednostki dyskowe 5,25", 360 KB DS/DD
- 1 łącze szeregowe RS 232 C
- 1 łącze drążka sterowniczego
- karta grafiki monochromatycznej 720x348 punktów
- monitor monochromatyczny 14" bursztynowy
- klawiatura 101-klawiszowa z polskimi znakami
- karta sterownika FDD
- zegar czasu rzeczywistego/kalendarz z podtrzymaniem bateryjnym
- dysk twardy 20 MB z kontrolerem i kablami



ZESPO…

współpracujące z mikrokomputerem EMIX 86 XT Turbo oraz innymi zgodnymi z IBM PC/XT/AT

- karta grafiki kolorowej
- karta grafiki monochromatycznej
- karta wielofunkcyjna I/O PLUS 2
- płyta systemowa z pamięcią 640 KB
- interfejs pomiarowy (IEC 625, HPIB, IEEE 488)
- karta sterowania dziurkarką i czytnikiem taśmy papierowej
- łącze szeregowe RS 232 C
- karta transmisji BSC
- karta transmisji 1200/300
- karta 4 x RS 232 C
- karta sterowania pamięcią taśmową PT-305 z oprogramowaniem (możliwość konwersji zbiorów IBM XT/AT ‹—› MERA 9150, IBM XT/AT ‹—› ODRA 1305)

KOOPERACJA

w zakresie montażu, starzenia i testowania pakietów elektronicznych

STOLIK

pod komputer, drukarkę i teleks z naturalnego drewna, ergonomiczny i estetyczny.

LOKALNA SIEĆ MIKROKOMPUTEROWA

EmNet

zbudowana na bazie mikrokomputerów EMIX 286 AT i EMIX 86 XT Turbo.
Pokazy i informacje w Biurze Technicznym firmy.

Ko-63/104/3

"Beta B."
AGENCJA
INFORMATYCZNA
41-200 SOSNOWIEC, P-254
oferuje:
PROGRAMY, INSTRUKCJE
DLA KOMPUTERÓW:
Acorn Amstrad
Commodore IBM
ATARI SHARP
Telef. 632-935, 690-385

Ko-97

WOJEWÓDZKIE
PRZEDSIĘBIORSTWO
HANDLU WEWNĘTRZNEGO
ODDZIAŁ W TYCHACH

VIDEOBIT

43-100 Tychy, aleja ZMP 77
tel. 27-69-75

Poleca dla j. g. u.:

- minikomputery 8-bitowe (Atari, Commodore, Schneider- Amstrad),
- minikomputery 16-bitowe kompatybilne z IBM PC,
- drukarki 10" i 15" firm STAR, EPSON, AMSTRAD,
- magnetowidy,
- kamery wideo,
- aparaturę badawczo-naukową.

Zapewniamy o atrakcyjnych cenach!

Ko-41 112 5

Firma

MUEL

oferuje do sprzedaży:

1. **Interfejs** do **ZX Spectrum, ZX Spectrum Plus, Timex 2048,** umożliwiający współpracę z czterema napędami dysków elastycznych, **RAM**-dyskiem, dowolną drukarką graficzną, maszyną do pisania **Robotron S-6120**, monitorem ekranowym, rozszerzający Basic oraz system operacyjny **ZX Spectrum.** Nie zajmuje pamięci **RAM!!!**
2. Sterowany ikonami programator **Eprom** 2716÷27256 do **ZX Spectrum.**
3. Przeróbkę drukarki DZM 180 na drukarkę graficzną.

Informacja: tel: 33-40-91
Korespondencja: MUEL
ul. Cząstkowska 30,
01-678 Warszawa
Zamówienia: Spółdzielnia
Rzemieślnicza
Specjalistyczna
Elektryków,
ul. Grójecka 128,
02-383 Warszawa
Wykonawca: MUEL.

Ko-15 120 4

Programy komputerowe, instrukcje i udoskonalenia techniczne
pocztą
dla ATARI, AMSTRADA, COMMODORA i IBM
wysyła
Agencja Mikrokomputerowa
Sosnowiec P-157, tel. 63-29-35

Ko 20 76 3



dataCo

Przedsiębiorstwo Wdrażania Postępu Technicznego
dataCo - Trading

01-710 Warszawa, ul. Włościańska 25
tel. 33-59-73 tlx 816159 datac pl

Posiada w sprzedaży następujące oprogramowanie dla mikrokomputerów kompatybilnych z IBM PC XT/AT:

- instalacja polskich liter (DOS),
- polski edytor tekstowy MS (DOS),
- biblioteka okien dla języka C (DOS, XENIX),
- system finansowo-księgowy (DOS, XENIX),
- system gospodarki materiałowej (DOS, XENIX),
- system płacowy (DOS),
- biblioteka graficzna do Turbo Pascala dla karty Hercules (DOS).

Jeżeli jesteś autorem oryginalnego programu aplikacyjnego - skontaktuj się z nami, będziemy pośredniczyć w sprzedaży Twojego programu dbając o ochronę Twoich praw autorskich!

Zapraszamy Naszych Klientów do Działu Oprogramowania
Warszawa, ul. Dzika 4 tel. 31-80-75

Ko 71 110 4

GLAD

BIURO USŁUG KOMPUTEROWYCH
AL. UJAZDOWSKIE 18/14
00-478 WARSZAWA
TEL. 28 01 76

PROPONUJEMY INSTRUKCJE OBSŁUGI I DOKUMENTACJĘ W JĘZYKU POLSKIM DLA KOMPUTERÓW:

AMSTRAD * PASCAL MT+ * NAUKA PROGRAMOWANIA W JĘZYKU MASZYNOWYM * SUPERCALC.
ATARI XL/XE * TURBO BASIC * LOGO cz.1: WSTĘP DO PROGRAMOWANIA * NAUKA PROGRAMOWANIA W ATARI BASIC * MAPA PAMIĘCI.
C 16 * JĘZYK MASZYNOWY * KURS BASICA.
C +4 * PODRĘCZNIK PROGRAMISTY * INSTRUKCJA OBSŁUGI.
C 64 * SAMOUCZEK * PODRĘCZNIK PROGRAMISTY.
MSX * INSTRUKCJE OBSŁUGI * POL-TXT * TURBO PASCAL.
SHARP * INSTRUKCJE OBSŁUGI DLA MZ-800, MZ-700 * ASEMBLER- -URAS.
IBM PC * smARTWORK wer. 2.7 pl * smARTWORK wer. 3.4 pl * ASEMBLERY: Z80, I8080/I8085, I8048/I8035.

PROWADZIMY SPRZEDAŻ WYSYŁKOWĄ
WYSTAWIAMY RACHUNKI

Ko 119 147 8

"MIKROSERWIS"

ul. Marusarzówny 6
80-288 GDAŃSK-MORENA D
tel. 48-50-63 godz. 9-17

poleca naprawy:

- komputerów **SPECTRUM, COMMODORE, AMSTRAD, IBM.**
- drukarek **STAR.**
- zasilaczy do **IBM.**

oraz cartridge do **C-64**
/FINAL II, DYSKOBOL/
C-16, C+4/UNIVERSAL/.

Ko-122/159 9

Zamierzasz kupić komputer, kserokopiarkę, fotokopiarkę...? Zadzwoń najpierw do nas!





Przedsiębiorstwo "Panda" działając z upoważnienia firmy **FARCOM INC. LTD.** informuje, że zakupiony wysyłkowo w tej firmie sprzęt, to jest:

- systemy komputerowe PC XT/, AT, RT, 8-34 MHz! w dowolnej konfiguracji i wersji (standard, baby, tower, portable, laptop) najlepszych firm światowych;
- urządzenia peryferyjne wszelkiego typu;
- kserokopiarki i fotokopiarki wraz z materiałami eksploatacyjnymi;
- urządzenia do odbioru telewizji satelitarnej;

objęty jest serwisem technicznym naszego przedsiębiorstwa w ramach rocznej gwarancji udzielanej przez FARCOM INC., LTD.

Przedsiębiorstwo "Panda" zapewnia:

- wykonanie przeglądu zerowego i testowanie sprzętu;
- wydanie użytkownikowi karty gwarancyjnej firmy FARCOM INC., LTD;
- wykonanie wszelkich zobowiązań z tytułu gwarancji;
- doradztwo techniczno-handlowe.

FARCOM to twój najlepszy wybór!!!

UWAGA! Każde zamówienie złożone do dnia 1989.03.15. bierze udział w **losowaniu** specjalnej **premii** promocyjnej - **wyjazdu do Taipei i Singapuru** wraz ze zwiedzaniem wystawy **Computex 89!**

Skontaktuj się z nami już dzisiaj...

PRZEDSIĘBIORSTWO TECHNICZNO-HANDLOWE OBSŁUGI IMPORTU I EKSPORTU „PANDA" sp. z o.o.
40-320 KATOWICE UL. 3 MAJA 23/7, TEL.: 586 633

...zadzwoń, napisz, przyjedź

Ko 195 187 11

Po historycznym podpisaniu licencji AMSTRAD - I. B. M.

POLANGLIA LTD

171-5 Uxbridge Road, LONDON W13 9AA
Tel: 840 1715; Fax: 840 7136; Tlx: 946581

Wyłączne przedst. na Polskę firmy **AMSTRAD**
oferuje NOWĄ GENERACJĘ komputerów

AMSTRAD PC2000

(licencjonowany przez IBM BIOS)

i MONITOTY V.G.A

Inauguracyjna oferta: **ok. 25% taniej**
od cen detalicznych w Wielkiej Brytanii
oraz rozwiązanie dla użytkowników PC w sieci:

"THE AMSTRAD NETWORK"

Również zgodne z IBM komputery AMSTRAD **PC 1640,** 1512 i przenośne **PPC**; Edytory tekstów AMSTRAD **PCW 9512, 8512 i 8256**; Komputery domowe AMSTRAD CPC 6128, 464, Sinclair Spectrum +2, +3 **NOWY SINCLAIR PC 200 (professional series - zgodny z IBM)** oraz **ATARI,** AMIGA, PSION, drukarki **STAR; REWELACYJNE DRUKARKI AMSTRAD** LQ5000di, 3500, DMP4000, 3250di, 2160, peryferia i **AMSTRAD VIDEOMATIC cameorder.**

PO NAJNIŻSZYCH CENACH W EUROPIE

C-61 196 10

PŁACE
KADRY
GOSPODA
MATERIA

Logic

PRZEDSIĘBIORSTWO WDRAŻANIA ZAAWANSOWANYCH TECHNOLOGII
ul. Wilcza 44 m 8, 00-679 Warszawa
tel. 28-37-30 tlx. 817322 logic pl

OFERUJE

- dowolny sprzęt PC
- drukarki, podtrzymywacze napięcia
- sieci
- systemy OS/2, XENIX, NOVELL
- programy na zamówienie
- galanterię komputerową

a ponadto

do Ventury Publisher (Rank Xerox)

- polskie i rosyjskie litery
- program dzielący polskie wyrazy
- polski i rosyjski edytor tekstów

Nasze komputery pracują m.in. w elektrowniach i szpitalach. Zapewniamy 12-miesięczny serwis gwarancyjny i serwis pogwarancyjny.

sp. z o. o.

Logic

NAWROT Peripherials & computer system

Jest firmą specjalizującą się w KOMPUTERACH I OPRZYRZĄDOWANIU

» PROGRAM NASZ OBEJMUJE «

KOMPUTERY IBM KOMPATYBILNE Z CAŁYM OPRZYRZĄDOWANIEM
CZĘŚCI ZAMIENNE CGA EGA
KARTY HERCULES, CGA, FGA itp.
TWARDE DYSKI SEAGATE, NEC
MONITORY
TERMINALE
PLOTERY: ROLANDA, G-GRAPHTEC
DIGITIZERY: ARISTOTAB, SUMMAGRAFIK
KOPIARKI: NASHUA, RANK XEROX

UKŁADY SCALONE: PAMIĘCI, MIKROPROCESORY itp.
DYSKIETKI: NASHUA, MAXELL, BASF, TDK, NO NAME
DRUKARKI FIRMY "STAR"
NX 15 – 730 DM, **ND 15** – 970 DM, **NR 15** – 1170 DM, **NB 24-15** – 1400 DM, **LC 10** = NX 1000 - 450 DM, **ND 10**, **SR 15**, **SR 10**.
RÓWNIEŻ FIRMY EPSON i NEC
SYSTEMY KOMPUTEROWE LO-NET FOX REASERCH ARC NET
DO WSZYSTKICH DRUKAREK POSIADAMY **TAŚMY BARWIĄCE**

Oferowane towary można nabywać

w naszym przedstawicielstwie:
DITMAR-KOEL-STRASSE 22 oraz BOTTMANNSTR 5
2000 HAMBURG 11 — 1000 BERLIN WEST 65
WEST GERMANY
TEL. 040/319 23 07
TELEX 2161853 ZAN D

lub poprzez wpłaty na konto bankowe:
NAWROT-IMPORT-EXPORT
DEUTSCHE BANK AG HAMBURG
BZL 200 700 00
KTO NR 39 70 399

W powyzszym przypadku nalezy listownie lub telefonicznie (9.00-19.00) złozyc zamowienie z podaniem dokładnego adresu odbiorcy. Koszta przelewu z konta oraz przesyłki pokrywa **wpłacający.**
Opłata za paczkę do 7 kg wynosi 20 DM. Za kazdy następny kg 2 DM + 5 DM od paczki. Np. koszt wysyłki drukarki SG 15 wynosi 35,-DM.
Nasze transporty do Polski wysyłane są co tydzien. **Gwarantujemy Państwu dostarczenie przesyłki w ciągu 2 tygodni** od momentu wpłynięcia przelewu na nasze konto.

NA ŻYCZENIE KLIENTÓW ZAŁATWIAMY ZAKUP I WYSYŁKĘ ELEMENTÓW ELEKTRONICZNYCH W ILOŚCIACH HURTOWYCH.

Ko-136-138/142-10

ELECTRONICS EXPORT

"ELECTRONICS EXPORT" PO.Box 869,London W5,ANGLIA-Tlx 8950511 oneone G (25190001 ref)
Tel (0-0441) 993 7000 - **Showroom i sklep** ; 19,Queens Parade,London W5,Ealing

ATARI

ATARI ST	£	$
520 STM+Drive SF 354(0,5 Mb)	250	433
520 STM+Drive SF 314(0,5 Mb)	275	476
520 STFM (wbud.drive 0,5 Mb)	275	476
520STFM(wb.dr.1Mb,22gry,joystik)	350	605
1040 STFM(z mod.TV) nowość	435	750
MEGA ST 2 MB (drive 1 Mb)	810	1399
Monitor mono SM124	135	233
Monitor SM124 (kupowany z ST)	95	165
Monitor kol,SC1224 (kup.z ST)	270	467
DRIVE ST 1 Mb,3½" NEC/CHINON	100	173
DRIVE ST 1 Mb,5¼" NEC/CHINON	125	217
DYSK SZTYWNY ST NEC 20 Mb	400	692
Monitor kolor m.r.PHILPS 8803	199	345
EMULATOR IBM (wymaga drive 5¼")	60	100
Zegar bateryjny ST	25	44
TANIEJ-ATARI LASER PACK	1600	2769

2Mb MEGA+SM124+SLM804 LASER PRINTER
Desktop Publisher+VIP profesional.

DYSKI-NASHUA,3 M,SKC		
3½" Bulk 25 pcs DSDD	25	43
3½" DSDD pakowane po 10	15	26
5¼" DSDD bulk 50 pcs	18	31
5¼" DSDD pakowane po 10	6	11
5¼" DSHD pakowane po 10	15	26
COMMODORE C64-KOMPLET	140	242

Komputer,datacorder C2N,joystick QH+10 gier sport.

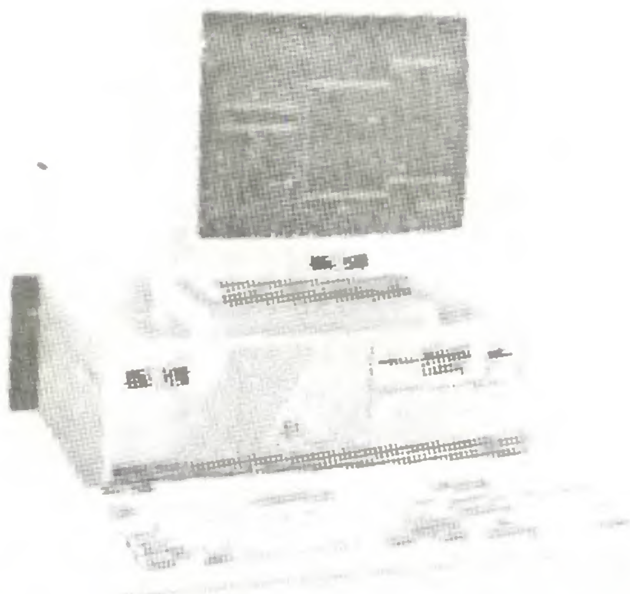


AUTOCOMPUTER CO. LTD

VIP XT/AT

Turbo 10 MHz, Phoenix/Award BIOS,8 gniazd,mala obudowa,gniazdo koprocesora,karta hercules,centronic klawiatura 101 klawiszy.Instrukcja ,opis.XT-procesor 8088-1,AT-procesor 80286-10,zegar/kalendarz, RS232,mozliwosc rozbudowy pamieci na karcie do 4 MB.,Funkcja"EMS".**Wszystkie** stacje dyskow NEC/ CHINON lub SEAGATE (H.D.).

	£	$
VIP 800/XT-1x360K,256K Ram	361	625
VIP 800/XT-1x360K,640K Ram	462	800
VIP 800/XT-2x360K,256K Ram	419	725
VIP 800/XT-2x360K,640K Ram	520	900
VIP 800/XT-360K,20Mb,256K Ram	607	1050
VIP 800/XT-360K,20Mb,640K Ram	708	1225
VIP 220/AT-1.2Mb,20Mb,512K Ram	838	1450
VIP 220/AT-1.2Mb,20Mb,1024KRam	982	1700
Monitor bursztynowy 12"	80	140
Monitor bursztynowy 14"	110	190
Monitor EGA 14"+ karta EGA	347	599
STREAMER WEWNĘTRZNY 40Mb	360	623
MYSZ "GENIUS 6" (RS232)	35	60



VIP XT/AT

Opus Technology.

OPUS XT/AT z monitorem

Renomowany produkt fmy angielskiej.

	£	$
PCIII/XT-360K,30Mb,1024KRam	£ 1095	1895
PCV/AT-1.2Mb,30Mb,1024KRam	1395	2413
PCIV/XT,EGA,30Mb,768KRam,360K.	995	1730

Najnowszy produkt fmy OPUS.Polecany !

System EGA do III i V-dopłata	300	519

DRUKARKI-NAJLEPSZE !

Centr/IBM,papier zwykły i komputerowy.

CITIZEN 120D,NLQ,120 z/sek	145	250
STAR LC10,NLQ,144 z/sek,4 font	195	337
STAR GEM-15x,120 z/sek,40 cm	115	200

Wyprzedaż.Niska cena.Ostatnie egzemplarze.

NEC P2200-24 igły,LQ,168 z/sek	280	485

Najlepsza i najtańsza drukarka 24 igłowa.
25 cm,8 fonts,orginał+2 kopie.

Wpłat można dokonywać w funtach ang. i dolarach amerykańskich.Do powyższych cen należy doliczyć **£ 5/$ 10** na koszty opakowania,ubezpieczenia.Wszystkie artykuły dostarczane są drogą lotniczą Warszawa Okecie.Koszt frachtu opłaca odbiorca w złotych*)
Zwyżki cen niektórych komputerów spowodowane są brakiem kości RAM na rynkach światowych.
Przegląd zerowy w Polsce,roczna gwarancja na części ,autoryzowany punkt serwisowy w Polsce.
"UNICOMP" 05-870 Błonie
ul.Przybysza 16.tel Wwa 554 554
Po zrobieniu przekazu telegraficznego na nasze konto w funtach lub dolarach,kopię dowodu wpłaty wysłać do nas listem poleconym wraz z zamówieniem na nasz adres.Jeżeli macie telefon dom/praca prosimy podać numer.Podać zawód odbiorcy przy zamówieniach ATARI i OPUS AT.Termin dostawy ATARI/ OPUS 3-4 tyg.,VIP ok 5 tyg.od otrzymania wpłaty....
Nasz bank:**BANK HANDLOWY SA.Oddział Londyn 4,Coleman Str,London EC2 . Rach No 200047**
*)Przybliżony koszt frachtu za komputer 16000 zł.

C 7 10

OGŁOSZENIA EKSPRESOWE

przyjmuje: Biuro Ogłoszeń, ul. Poznańska 38, 00 - 689 Warszawa. tel. 28 23 09

Giełda

Giełda w ZSRR

"Tego jeszcze nie było !" - te słowa padają często w rozmowach o obecnej sytuacji w Związku Radzieckim. Nie było również w "Giełdzie" wzmianek o rynku mikrokomputerowym w ZSRR, pora więc wspomnieć o tym interesującym skądinąd zjawisku.

Opisywałem już jakiś czas temu moją wizytę w salonie "Elektronika" w Moskwie. Sprzedawano tam wówczas komputer domowy BK 0010.01. Od tamtej chwili oferta poszerzyła się o kilka nowych modeli. Wszystkie proponowane maszyny to komputery szesnastobitowe z procesorem zgodnym z DEC LSI 11. W ciągłej sprzedaży znajduje się wciąż BK 0010. Ze względu na znaczne zainteresowanie klientów nowym towarem prowadzone są zapisy. Chętne na zakup mikrokomputerów są w większości organizacje, np. oświatowe lub powstające jak grzyby po deszczu "kooperatiwy" (coś pośredniego między prywatną inicjatywą a spółdzielnią). Zdarzają się również osoby prywatne, choć nie każdy może sobie pozwolić na tak duży wydatek. A oto ceny w salonie ***"Elektronika"*** w Moskwie:

BK 0010 (16b, 32K RAM, 24K ROM, foliowa klawiatura, interpreter języka FOKAL)	600 Rbl
BK 0010.01 (jw.,32K ROM, wbud. BASIC, klawiatura kontaktronowa)	650 Rbl
UKNC (komputer osobisty; dwa procesory 16b, 192K RAM, dobra klawiatura, 2 napędy dysków 5,25", monitor monochromatyczny, drukarka ROBOTRON)	7746 Rbl
UKNC jak wyżej, wysokiej klasy monitor kolorowy	9324 Rbl
DWK 4 (komputer profesjonalny, 1M RAM, 2 napędy dysków 5,25" 720K, dysk twardy 5MB, monitor kolorowy, drukarka D100, ploter)	27024 Rbl
Zestaw edukacyjny (jednostka nauczycielska DWK 2M, 12 mikrokomputerów BK 0010SZ, 12 monitorów monochromatycznych, 12 interfejsów sieci lokalnej)	30000 Rbl
Kaseta C 60 z programami dla BK 0010	40 Rbl
"Mikrosza" (to już w innym sklepie; mikrokomputer z procesorem 8080, 32K RAM, 2K ROM, interpreter języka BASIC wczytywany z kasety)	500 Rbl
"Sura" (8080, 64K RAM, 16K ROM, wzorowany na MSX)	880 Rbl

Sprzęt komputerowy sprzedają również komisy, przy czym ciekawostka: prawo pierwokupu np. drukarek mają instytucje państwowe.

Istnieje oczywiście rynek nieoficjalny (zorganizowanych giełd - w naszym rozumieniu - nie ma), ale jest to działalność prywatna, prawnie nie sankcjonowana. Jest to jak dotąd najlepszy sposób kupienia sprzętu znanych i renomowanych firm światowych.

Kilka przykładów cen:

ATARI 520 ST + SM 125 + SF 354	6000-7000 Rbl
ATARI 1040 STF + SC1224	7500-9000 Rbl
monitor mono SM 125	1500 Rbl
monitor kolor SC 1224	2300-2700 Rbl
ATARI XL/XE (zwykle z PEWEX-u...)	1200-1300 Rbl
ATARI 130 XE + XC 12	1600-1700 Rbl
magnetofon ATARI	300-400 Rbl
stacja dysków ATARI 1050	1600-1800 Rbl
stacja dysków LDW 2000	1900-2100 Rbl
drukarka ATARI 1029	1800-2000 Rbl
drukarka EPSON, NLQ	2400-2700 Rbl
COMMODORE C128 z monitorem (komis)	6000 Rbl
AMIGA 500 + monitor kolor (brak programów...)	5500 Rbl
AMSTRAD PC 1512 DD mono	5500 Rbl
PC/XT SD + HD 20 MB, CGA	11000-12000 Rbl
monitor EGA	3000 Rbl
napęd dyskowy NEC 5,25", 360 KB	400 Rbl
napęd dyskowy NEC 5,25", 720 KB	700-800 Rbl
kabel do drukarki typu CENTRONICS	80-100 Rbl
dyskietki 5,25", firmowe, zapieczętowane, 10 szt.	110-120 Rbl
jak wyżej, otwarte	80-100 Rbl
dyskietki 3,5" za 1 szt.	15-35 Rbl

Dla porównania:

chleb	0.25-0.40 Rbl
mleko, 1 litr	0.36-0.50 Rbl
telewizor kolorowy	645 Rbl
samochód Łada Samara	9000 Rbl
średnia pensja	200 Rbl

Najpopularniejszym zachodnim mikrokomputerem w ZSRR wydaje się obecnie być ośmiobitowe Atari (najczęściej "importowane" zza Buga). Daje się zauważyć gwałtowny wzrost popularności komputera Atari ST, który uchodzi za wzór komputera dla ludzi twórczych. Poszukiwane są stacje dysków SF 314 oraz 5,25 cala do Atari ST a także dyskietki 3,5 cala.

Notował: Michał Setlak

Ceny na „Softargu'88"

Wrażenia z Ogólnopolskich Targów Oprogramowania Softarg'88 obszerniej opiszę za miesiąc, dziś kilka słów o tym, co dla kupca najważniejsze - cenach na tym rynku..

Jeśli Twoja firma ma komputer klasy PC/XT/AT lub pragnie go nabyć, to zajęcie dlań oferuje ponad 150 obecnych w Katowicach firm, a ceny na tym jarmarku nawet w dobie inflacji stale spadają.. Dobry przykład dają konkurencji weterani - w firmie CSK wielka wyprzedaż:

PL-Tekst staniał z 450 do 199 tys. zł

BGraf z 480 do 299. tys.,

TRys ze 160 do 99 tys. zł.

Widać wpływ przyszłej strefy wolnocłowej w Gdyni na zamiłowanie okolicznych przedsiębiorców do pary cyfr 99.

Najtańszym produktem oferowanym na Targach, a zarazem największym ich przebojem handlowym był kosztujący 40 tys. zł słownik angielsko-polski oferowany przez p. Jacka Skalmierskiego z Gliwic - tłumaczył on wprowadzany tekst słowo po słowie, proponując obszerne wyjaśnienia leksykalne poszczególnych terminów.

Najdroższy program trudno wskazać, gdyż wielu dostawców preferowało wciąż ceny umowne, tzn. wedle oceny zasobności klienta..

Dla redaktora najciekawsze były notowania jego elektronicznej konkurencji, a więc programów typu "pulpit wydawniczy": program CYFROSET 1.0 firmy CYFRONEX można nabyć już za 1.3 mln zł, a program PL-DRUK firmy Micrograf za 1.5 mln zł. Na zestaw sprzętowy złożony z komputera, twardego dysku i drukarki laserowej trzeba przygotować dodatkowo co najmniej 12.7 mln zł (maksymalna proponowana konfiguracja wymaga już 30 mln).

Oba te programy są oryginalnymi polskimi opracowaniami, jeśli jednak ktoś otrzyma od wujka z Ameryki standardy światowe, może ograniczyć się do zakupienia w firmie MicroVAC Rolanda Wacławka programów adaptujących MS-Windows i PageMaker do pracy w języku polskim za jedne 300 tys. zł wraz z obszernymi instrukcjami, plus koszt wyrzutów sumienia, jeśli wujek do przesyłki nie dołączył licencji użytkownika od firm Microsoft i Aldus.

Gazetki gazetkami, ale o pozycji firmy decydują finanse. Programy finansowo-księgowe, kadrowo-płacowe oraz obsługujące gospodarkę materiałową i magazynową stanowiły ponad połowę targowej oferty, stąd możemy przedstawić tylko nieliczne z nich. Najpełniejszy ich zestaw, nagrodzony przez targowe jury, przedstawiła firma InterAms: Ceny składających się nań kilkunastu programów wahały się od 290 tys. (system DYSCYPLINA wersja 1.0 - dyscyplina w Polsce tradycyjnie jest nisko ceniona) do 4.900 tys. (system planowania operatywnego i system technicznego przygotowania produkcji). W sumie średniej wielkości przedsiębiorstwo chcąc jednym skokiem trafić do komputerowego raju powinno mieć w zanadrzu ok. 70-80 mln zł na sprzęt i ok. 25 mln zł na oprogramowanie. Najbardziej umiarkowana wersja kosztować musi ok. 6 mln zł, w tym oprogramowanie ok. 500 tys - tyle kosztują programy firmy Macrosoft, która otrzymała nagrodę targową za generator aplikacji.

Targi ukazały kulturotwórczą moc biurokracji: pojawiła się nowa, czysto polska kategoria oprogramowania: kilkanaście programów przeznaczonych wyłącznie do obliczania podatku od ponadnormatywnych wynagrodzeń (PPWW). Spółka BISTER za jedne 250 tys. oferowała program, który pozwala w razie potrzeby wstecz zmienić formułę naliczania tego podatku. Po raz pierwszy koszt zakupu komputera niektórym zwraca się za pierwszym jego uruchomieniem...

O cenach innych programów finansowo-biurowo-administracyjnych oraz oprogramowania inżynierskiego - wkrótce.

Władysław Majewski.

skład: LogoScript Sp. z o.o.